Face verification based on Support Vector Machines

Maria Trias

June 1, 2005

Contents

List of Figures

Abstract

Face verification based on facial images taken in controlled or uncontrolled scenes is a technique that has became popular in the last years for security applications. Verification systems task is, basically, to check if an image matches with the model related to the identity stored in the database, so that the system decides if the person of the picture is a client or an impostor.

For this, statistical models of appearance (PCA, ICA, LDA,...) that can encode the shape and texture of the face in a more compact parameter vector are becoming popular, and then techniques, that come from a simple distance-based classifiers to a complex SVM classifier, are used to match the new images with the models stored

The goal of this project is to build and evaluate such verification approaches on classical databases, such as BANCA.

Chapter 1

Introduction

In the recent years, lots of scenes related to security have appeared in our life without realizing it. We use a password to log in a computer and access to Internet or a PIN number to electronically pay. Although, within the last several years, the use of biometric technologies, which includes face recognition and verification, in a wide range of applications have increased.

One of the main applications of biometric technologies is security. Since the terrorism actions of 11th September in New York ,the attitude about security has changed in the world. Now, everybody is aware of the last changes in security in the check-in in the airports or even in the train stations. While in the past a passenger of a flight from USA had only to pass a simple detector before the boarding, now a system of face recognition and verification is used in most cases. In spite of being one of the most known examples of biometric technology applications there are much more as shown in table 1.1

There are two main reasons for this tendency of using more and more biometric technologies; on one hand we have the recently emerged growing trend towards some applications as e-commerce, teleworking and e-banking, and, on the other hand, the decrease of the cost of biometric sensors and processors, so that, after 30 years of research, biometric technologies are feasible.

While some of the biometric technologies require a cooperation of the client in order to be used in security applications (fingerprint analysis, retinal or iris scans...) the analysis of frontal or profile images of the face is often effective without the participant's cooperation. This is what makes face recognition and verification to be a very useful tool in some security scenes.

Therefore, face recognition and verification has been presented lately as

Areas	Specific Applications
Entertainment	Videogame, virtual reality, training programs Human-robot interaction, human-computer interaction
Smart Cards	Driver's licenses, entitlement programs Immigration, national ID, passports, voter registration Welfare fraud
Information Security	TV parental control, personal device log-on, desktop logon Application security, database security, file encryption Intranet security, internet access, medical records Secure trading terminals
Law enforcement and surveillance	Advanced video surveillance, postvent analysis Shoptlifting, suspect tracking and investigation

Table 1.1: *Typical Applications of Face Recognition*

one of the most successful applications of biometric technology and image analysis. Most researchers from different disciplines, such as pattern recognition,computer vision, computer graphics and more have been attracted by this topic. This is evidenced by the emergence of face recognition conferences such as International Conference on Audio and Video-based Authentification (AVBPA) since 1997 and the International Conference on Automatic Face and Gesture Recognition (AFGR) since 1995, systematic empirical evaluations of face recognition techniques (FRT), including the BANCA or the FERET protocols, and many commercially available systems.

Immerse in this line of research,the main goal of this project is, therefore, to build a recognition and verification system, using a SVM classifier, and test it on the classical databases, especially BANCA [1]

[1]BANCA[1] is a multi-modal database intended for training and testing multi-modal verification systems. In this database there are 208 people, half men and half women recorded in three different scenarios, controlled, degraded and adverse for four different languages (English, French,Spanish and Italian). This database is furthered described in Annex A banca[1].

1.1 Problem Overview

Although the general problem of face recognition and verification can be formulated in the same simple way (given some images from a person, we recognize/verify his identity by using a stored database of faces), there are some differences that we should point out. While in recognition, given an image of a person we decide the identity of him; in verification, given an image of a person, who claims an identity, we should authenticate him, that means that we should decide whether his identity is the claimed one or he is an impostor.

The solution to these problems involve segmentation of the faces (face detection) from cluttered scenes, feature extraction from the face regions and, finally, the recognition/verification takes place. This process is shown more detailed in the figure 1.1

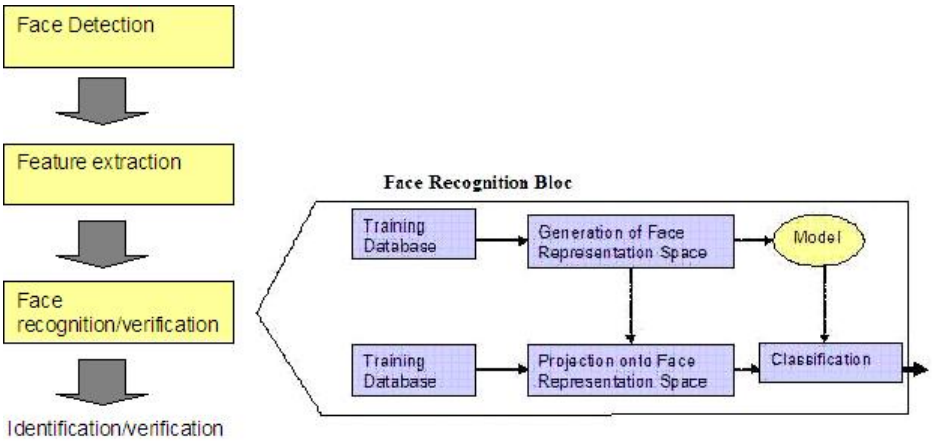


Figure 1.1: *Configuration of a generic face recognition/verification system*

The localization and detection step consists of detecting the faces and localize its position precisely in an image or video sequence. In the eventual step of features extraction the emotion recognition can take place.

And finally, in the third step, which this project will focus on, recognition or verification takes place. Such a task can be decomposed in two steps: finding an adapted face representation space and the classification, which performs the decision.

Complexity of the task

The complexity of this process comes from the difficulties that we find, and we have to face, in each of those three steps. However, the main handicap resides in the fact that we work with a deformable object, we work with faces that can vary depending on the facial expressions, the head's position, the lightening conditions or if the person we want to recognize/verify wears glasses or beard in some situations. The differences are evidenced in the figure 1.2.

That changeability makes our task more challenging. There are several strategies used to overcome those problems derived from the face deformability one of them is to discard those parts of the face which present more intra-class variance as LDA does.

Anyway, even doing with those strategies, it is not always possible to take the correct decision because the appearance of an image can strongly vary between two pictures of different people, or even between two pictures of the same person. There are several reasons for this variability, for example the difference between two people, the different expressions, different positions, the quality of the image or even those objects which can mask the main features of a face, like beard, glasses or moustache.

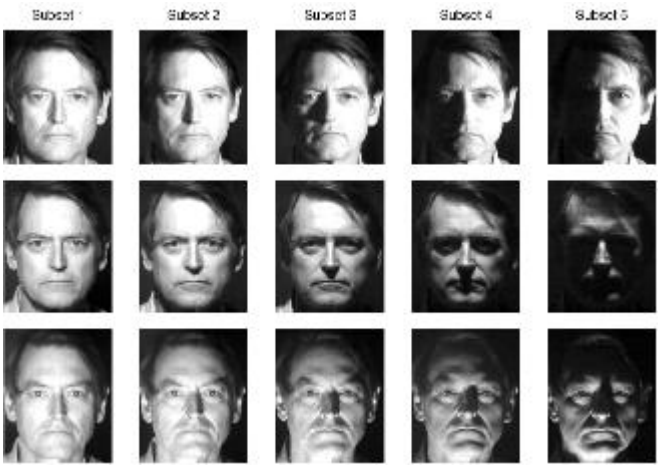


Figure 1.2: *Image of a person under different conditions*

1.2 Thesis organization

In order to reach our objectives we have used several techniques, which are going to be explained in detail below, as well as the reason that motivated us to choose them and not others.

In chapter 2, we will give a short State of the Art, and we will compare the main techniques used to create a Face Representation Space and to classify the samples in order to perform the classification. In chapter 3 we will explain which of that techniques we have chosen for our algorithm and in chapter 4 we will show the results obtained with such an algorithm. Finally, in chapter 5 there are some conclusions about the work done and the future perspectives of the research.

Chapter 2

State of the Art

While the task of a recognition system is to find the identity of a person using a database, the task of the verification system is to check if a test image matches with the model related to the identity stored in the database so that a person claiming a given identity is classified as client or impostor.

For this, many methods have been proposed during the past 30 years. Face recognition is such a challenging interesting problem that has attracted researchers from different backgrounds, that is the reason why the literature on face recognition is vast and diverse. Often, there are different principles applied in a simple recognition system, so that it is difficult to classify all of these techniques. In the literature the are several categorizations of them; one of the most interesting classifications is the one which is based on the study about how the human used both statistical models of appearance and local features for recognition. According to this classification, we have three different categories:

1. *Statistical models of appearance.* These methods can synthesize both face shape and texture, are becoming popular as they can encode a face in a relatively compact parameter vector and fast algorithms have been developed for matching such models to new images.

2. *Featured-based (structural) matching methods.* In these methods local features such as eyes, nose and mouth are first extracted and their locations and local statistics (geometric and/or appearance) are fed into a structural classifier.

3. *Hybrid methods.* Just as the human perception system uses both local features and the whole face region to recognize a face, a machine recognition system should use both.

However, there are other possible categorizations that are possible to use if we want to classify the different principles used in face recognition or verification. For example, we can differentiate which of them are used in each of the steps of the process (figure 2.1): the ones used to reduce the dimensionality, and the ones used to classify.

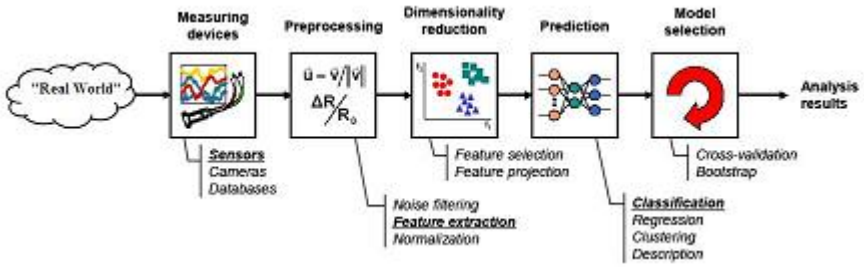


Figure 2.1: *Face recognition/verification system*

In this State of the Art, we will present several techniques classified as it is shown in figure 2.2

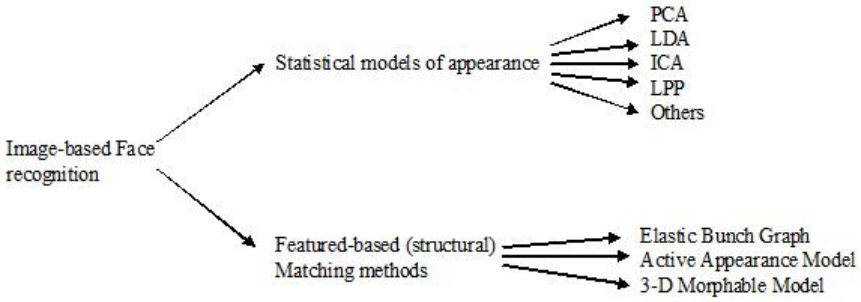


Figure 2.2: *Classification of the different Face recognition/verification methods*

2.1 Face Representation Space

The first step in the statistical approaches is to define a face representation space adapted for the identity classification in order to reduce the dimensionality of the problem. For this, two approaches are available:

- Feature extraction: creating a subset of new features by combinations of the existing features.

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} \longrightarrow \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{pmatrix} = f \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_M \end{pmatrix}$$

- Feature selection: choosing a subset of all the features (the ones more informative) $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} \longrightarrow \begin{pmatrix} x_i 1 \\ x_i 2 \\ \vdots \\ x_i M \end{pmatrix}$

The problem of feature extraction can be stated as:

- Given a feature space $x_i \epsilon R^N$ find a mapping $y = f(x) : R^N \rightarrow R^M$ with M<N such that the transformed feature vector $y_i \epsilon R^M$ preserves (most of) the information or structure in R^N.
- An optimal mapping $y = f(x)$ will be one that results in no increase in the minimum probability of error. This is, a Bayes decision rule applied to the initial space R^N and to the reduced space R^M yield the same classification rate.

In general, the optimal mapping $y = f(x)$ will be a non-linear function. However, there is no systematic way to generate non-linear transforms. The selection of a particular subset of transforms is problem dependent, for this reason, feature extraction is commonly limited to linear transforms: $y = Wx$ where y is a linear projection of x and the columns of W characterize the Face Representation Space:

Figure 2.3: *The same person under different lighting conditions*

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} \longrightarrow \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1N} \\ w_{21} & w_{22} & \dots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M1} & w_{M2} & \dots & w_{MN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}$$

Within the ground of linear feature extraction, two techniques are commonly used Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) which have been used successfully in the last decades. While PCA uses a signal representation criterion, Linear Discriminant Analysis (LDA) uses a signal classification criterion.

The representation should emphasize on the features that change from one identity to another. Then, some classification techniques are used to build the identity models.

However, in this process of decorrelating our images, it is possible that we extract some information we do not need in the process of deciding the identity. Sometimes, having that extra information could imply getting worse results. One example of this can be the lighting variation which is the main problem when we try to find the best Face Representation Space.

While much progress has been made toward recognizing faces under small variations of lighting, facial expression and pose, reliable techniques for recognition under more extreme variations have proven elusive. Lighting variations in the pictures taken is one of the factors that could most influence the results. This can be observed easily in figure 2.3 while both pictures belong to the same person, they can appear really different.

Depending on which technique we use, we can get some improvements in these scenes of variability. In the sections below, we will compare four of this techniques and how they face the problem of lighting variation.

2.1.1 Eigenfaces (PCA)

PCA is a technique commonly used for dimensionality reduction. PCA techniques, also known as Karhunen-Loeve methods, choose a dimensionality reducing linear projection that maximazes the scatter of all projected samples, or said in other words, it finds those vectors, W that best account for the distribution of face images within the entire image space.

Once we have those vectors that represent the face representation space, we perform the projection so that we transform a n-dimensional space to an m-dimensional space, beeing $n > m$:

$$y = W^T x$$

Since this is one of the techniques we have tested, it is further explained in chapter 3.

2.1.2 Independent Component Analysis (ICA)

ICA is a data analysis tool derived from the "source separation" signal processing techniques. The aim of source separation is to recover original signals S_i, from known observations X_j, where each observation is an (unknown) mixture of the original signals. So that, if A is the unknown mixing matrix, then the mixing model is written as

$$X = AS \tag{2.1}$$

Under the assumption that the original signals S_i are statistically independent, and under mild conditions on the mixture, it is possible to recover the original signals from the observations. The algorithmic techniques making this task possible are often called ICA, as they factories the observations as a combination of original sources. If we call the estimations of the original signals U and W the separating matrix, then we can obtain the mathematical expression for the process shown in figure 2.4:

$$U = WAS \tag{2.2}$$

If the mixing is linear, ICA estimates the inverse of the mixing matrix. The number of observations $N_1 \leqslant j \leqslant N$ must be at least equal to the number of original signals $M_1 \leqslant j \leqslant M$; often it is assumed that N = M. It is not necessary to have signals X_j to consider using ICA: X_j may also be multi-dimensional data (vectors). Assuming that each X_j is an unknown, different combination of original "source vectors" S_i, ICA will expand each

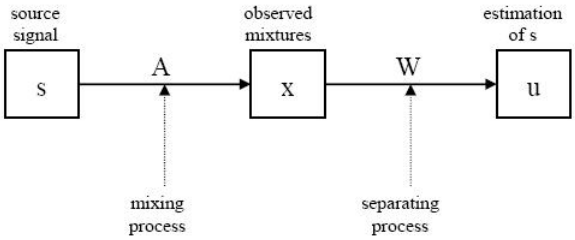


Figure 2.4: *Blind source separating model.*

signal X_j into a weighted sum of source vectors S_i (ICA estimates both the source vectors S_i and the coefficients of the weighted sum).

ICA can be viewed as a generalization of PCA. As previously discussed, PCA decorrelates the training data so that the sample covariance of the training data is zero. Whiteness is a stronger constraint that requires both decorrelation and unit variance. The whitening transform can be determined as $D^{-1/2}W^T$ where D is the diagonal matrix of the eigenvalues and W is the matrix of orthogonal eigenvectors of the sample covariance matrix.

Applying whitening to observed mixtures, however, results in the source signal only up to an orthogonal transformation. ICA goes one step further so that it transforms the whitened data into a set of statistically independent signals.

Thus, if ICA can be viewed as a generalization of PCA, we can think that the performance reached with ICA and PCA could be similar, but then, why this technique is not as popular as PCA? The answer is that while in PCA we chose the dimension that perfoms 99% of the total variation, there is no specific criterion to chose the dimension in ICA.

2.1.3 Fisherfaces (FLD)

PCA and ICA construct the face space without using the face class information. However, LDA finds an efficient way of representing the face vector space.

The Fisherface algorithm is derived from Fisher Linear Discriminant. By defining different classes with different statistics, the images in the learning set are divided into corresponding classes. Then some techniques similar to those used in the Eigenface method are applied.

This method is explained in more detail in chapter 3.

2.1.4 Laplacianfaces

Laplacian faces [2] is a new approach to face analysis which explicity considers the manifold structure (modelled by a nearest-neighbor graph which preserves the local structure of the image space).

A face subspace is obtained by Locality Preserving Projections (LPP). Each face image in the image space is mapped to a low-dimensional face subspace, which is characterized by a set of feature images, called Laplacianfaces. That face subspace preserves the local structure and seems to have more discriminating power than PCA.

The objective function of LPP is

$$\min \sum_{ij} (y_i - y_j)^2 S_{ij}$$

, where y_i is the one-dimensional representation of x_i and the matrix S is a similarity matrix. We can define S as follows

$$S_{ij} = \begin{cases} \exp(\frac{-\|x_i - x_j\|^2}{t}), \|x_i - x_j\|^2 < \varepsilon \\ \\ 0, otherwise \end{cases} \tag{2.3}$$

where ε is sufficient small, and $\varepsilon > 0$. We can say that ε defines the radius of the local neighborhood, or in other words, it defines the "locality". The objective function with the choice of symmetric weights ($S_{ij} = S_{ji}$)incurs a heavy penalty if neighboring points x_i and x_j are mapped far apart (i.e if $(y_i - y_j)^2$ is large). That means that minimizing it is an attempt to ensure that x_i and x_j are close and consequently the same for y_i and y_i.

Here is the sum up of the algebraic steps done in order to reach those objectives:

$$\frac{1}{2} \sum_{ij} (y_i - y_j)^2 S_{ij} = w^T X L X^T w \tag{2.4}$$

where $X = [x_1, ..., x_n]$, $y = w^T x$, D is a diagonal matrix (its entries are column -or row, since S is symmetric- sums of S: $D = \sum_{ij} S_i j$) and L=D-S is the Laplacian matrix. Matrix D provides a natural measure on the data points. The bigger the value D_{ii} (corresponding to y_i) is the more important is yi. Therefore, it is necessary to impose the constraint $y^T D y = 1$ or, which is the same,$w^T X L X^T w = 1$.

Finally, the minimization problem reduces to finding

$$\arg\min_{w}(w^T X L X^T w) \tag{2.5}$$

which accomplishes the constraint.

The transformation vector,w, that minimizes the objective function is given by the minimum eigenvalue solution to the generalized eigenvalue problem:

$$\underbrace{XLX^T}_{symmetr.\&posit.} w = \lambda \underbrace{XDX^T}_{symmetr.\&posit.} w \tag{2.6}$$

So that, when we finally have the eigenvectors which form an orthogonal base of the LPP, obtaining the Laplacianfaces conversion matrix is obvious:

$$W = W_{LPP} W_{PCA} \tag{2.7}$$

Comparison to Eigenfaces and Fisherfaces

Some experiments done by He et. al.[2] show that Laplacianfaces perform better than the original image space.

Besides, Laplacianfaces also perform better than Eigenfaces and Fisherfaces. It is experimentally demonstrated that the algorithm performs better when having frontal pictures of the faces, and, moreover, it takes advantage of more training samples.

Comparing such algorithm with Eigenfaces, it takes more discriminating information in the low-dimensional face subspace by preserving local structure which is more important than the global structure for classification, especially when nearest-neighbor classifiers are used.

While LDA can not work using just one training sample for each subject (since the within-class scatter matrix becomes a zero-matrix), with LPP it is possible to construct a complete graph and use inner products as its weights. In this case LPP can give the same result as PCA.

2.1.5 Feature-based methods of Face Representation

Bunch Graph

Elastic Bunch Graph Matching recognises faces by matching the probe set represented as the input face graphs, to the gallery set that is represented as

the model face graph. Fundamental to the Elastic Bunch Graph Matching is the concept of nodes. Essentially, each node of the input face graph is represented by a specific feature point of the face. For example, a node represents an eye and another node represents the nose and the concept continues for representing the other face features. Therefore the nodes for the input face graph are interconnected to form a graph like data structure which is fitted to the shape of the face as illustrated in figure 2.1.5.

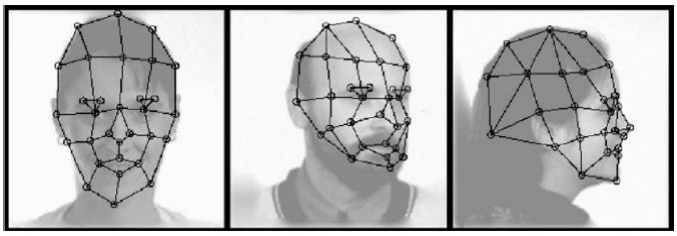


Figure 2.5: *Bunch Graph [3]*

In contrast, the model face graph represents the gallery set only used one model face graph to represent the entire gallery set. This is shown in figure 2.1.5 However, the model face graph can be conceptually thought of as a number of input face graphs stacked on top of each other and concatenated to form one model face graph, with the exception that this is applied to the gallery set instead of the probe set.

Therefore, this would allow the grouping of the same types of face features from different individuals. For example, the eyes of different individuals could be grouped together to form the eye feature point for the model face graph and the noses of different individuals can be grouped together to form the nose feature point for the model face graph.

Given the definition for the input face graph and model face graph, to determine the identity for the input face graph is to achieve the smallest distance in relation to the the model face graph for a particular gallery face. The distance is determined by the node similarity measure for the input face graphs to the model face graph, (ie. in [3]).

Active Appearance Model (AAM)

The AAM is an integrated statistical model which combines a model of shape variation with a model of appearance variations in a shape-normalized frame.

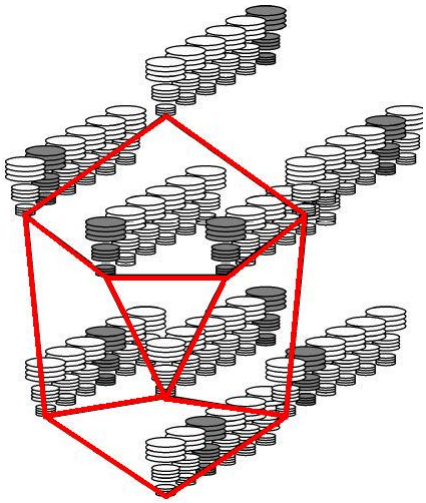


Figure 2.6: *The Face Bunch Graph (FBG) serves as a general representation of faces. Each stack of discs represents a jet. From a bunch of jets attached to a single node only the best fitting one is selected for a match, indicated by gray shading.*

An AAM contains a statistical model of the shape and gray-level appearance of the object of interest. It is constructed using a training set of labelled images of faces where some points mark the main features, as it is shown in figure 2.1.5

Figure 2.7: *The training image is split into a shape and shape-normalized texture.*

In order to model AAM subspace we consider that the training set is given as $(s; g)$ where a shape $s = ((x_1; y_1); :::; (x_K; y_K)) \in R^{2K}$ is a sequence of K points in the 2D image plane, and a texture g is the gray-level of the pixels of s.

The AAM shape subspace is trained by PCA analysis on the tangent

shape space

$$s = \bar{s} + P_s b_s \tag{2.8}$$

where P_s is the matrix of the k principal orthogonal modes of variation in s, which is a shape vector, and $\bar{s}$ is the mean shape.

To construct the appearance model, the image is warped to make control points match the mean shape. Then, the warped image region covered by the mean shape is sampled to extract gray-level intensity (texture) information. Such an information is represented with $g = (I_1, ..., I_m)$. Then PCA is applied to construct a linear model $g = \bar{g} + P_g b_g$, where $\bar{g}$ is the mean appearance vector, P_g is a set of orthogonal modes of gray-level variation and b_g is a set of gray-level model parameters.

We can express the combined model as

$$b = \begin{pmatrix} W_s b_s \\ b_s \end{pmatrix} = \begin{pmatrix} W_s P_s^T (s - \bar{s}) \\ P_g^T (g - \bar{g}) \end{pmatrix} \tag{2.9}$$

where W_s is a diagonal matrix of weights for each shape parameter. PCA is applied also to b $b = Qc$, where c is the vector of parameters for the combined model.

Given a new image and constructed model, the metric used in the decision step is $\Delta = |\delta I|^2$, where δI is the vector of intensity differences between the image given and the model.

2.1.6 Other dimensionality reduction methods

In the previous sections some methods to reduce the dimensionality are explained, but there are much more. In this section two more methods are explained not in detail, but just to have an idea of the wide range of possibilities there are in this field.

Exploratory Projection Pursuit (Friedman and Tukey)

EPP seeks an M-dimensional (M=2,3 typically) linear projection of the data that maximizes a measure of "interestingness". Interestingness is measured as departure from multivariate normality. This measure is not the variance and is commonly scale-free. In most proposals it is also affine invariant, so it does not depend on correlations between features . [4]

In other words, EPP seeks projections that separate clusters as much as possible and keeps these clusters compact, a similar criterion as Fisher's, but EPP does not use class labels.

Sammon's Non-linear Mapping (Sammon)

This method seeks a mapping onto an M-dimensional space that preserves the inter-point distances of the original N-dimensional space.

This is accomplished by minimizing the following objective function

$$E(d, d') = \sum_{i \neq j} \frac{[d(P_i, P_j) - d(P'_i, P'_j)]^2}{d(P_i, P_j)} \tag{2.10}$$

The original method did not obtain an explicit mapping but only a lookup table for the elements in the training set. Recent implementations using artificial neural networks (MLPs and RBFs) do provide an explicit mapping for test data and also consider cost functions (Neuroscale).

Sammon's mapping is closely related to Multi-Dimensional Scaling (MDS), a family of multivariate statistical methods commonly used in the social sciences.

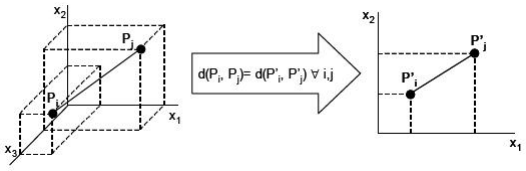


Figure 2.8: *Sammon's nonlinear mapping. Dimensionality reduction from 3D to 2D*

2.2 Classification

2.2.1 Statistical Classification

The most simplest way of clustering some samples is using statistical classifiers. There is a wide range of them: from the so called k-means algorithms to the hierarchical clustering algorithms.

Measures of similarity and dissimilarity: Metrics

The first step to explain these methods is to define what do we mean when we talk about "metric": A measuring rule d(x,y) for the distance between two vectors x and y is considered a metric if it satisfies the following properties.

$$d(x,y) \geqslant d_o$$
$$d(x,y) = d_o if x = y$$
$$d(x,y) = d(y,x)$$
$$d(x,y) \leqslant d(x,z) + d(z,y)$$

If the metric has the property $d(ax,ay) = |a|d(x,y)$ it is called a norm and denoted $d(x,y) = ||x-y||$.

The most general form of distance metric is *the power norm*

$$\|x-y\|_{p/r} = (\sum_{i=1}^{D} |x_i - y_i|^p)^{1/r} \tag{2.11}$$

where the parameter p controls the weight placed on any dimension dissimilarity, and the parameter r controls the distance growth of patterns that are further apart.

The most commonly used metrics are derived from the definition of the power norm:

Euclidean norm.

$$\|x-y\|_e = (\sum_{i=1}^{D} |x_i - y_i|^2)^{1/2} \tag{2.12}$$

Minkowski metric.

$$\|x-y\|_k = (\sum_{i=1}^{D} |x_i - y_i|^k)^{1/k} \tag{2.13}$$

Manhattan or city-block distance.

$$\|x-y\|_{c-b} = \sum_{i=1}^{D} |x_i - y_i| \tag{2.14}$$

Chebyshev distance.

$$\|x-y\|_{c} = \max_{1 \leqslant i \leqslant D} |x_i - y_i| \tag{2.15}$$

But other metrics which do not derive from the definition of power norm are also popular, for example,

the Quadratic distance, defined as

$$d(x,y) = \sqrt{(x-y)^T B (x-y)} \tag{2.16}$$

and *Mahalanobis distance*, which is a particular case of the last one:

$$d(x,y) = ((x-y)^T C^{-1} (x-y))^{1/2} \tag{2.17}$$

where C is the covariant matrix. If C is the identity matrix, then, Mahalanobis distance is equal to Euclidean distance.

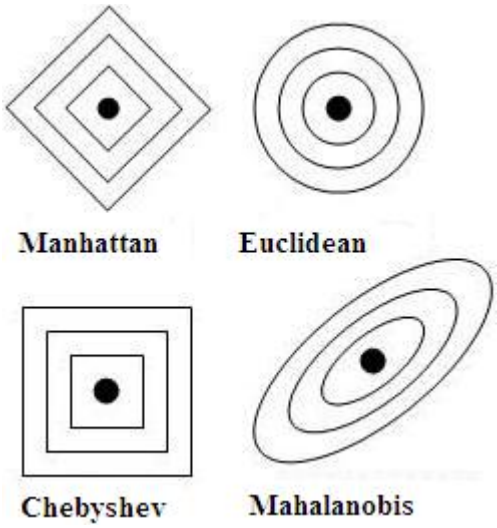


Figure 2.9: *The different contours for constant Manhattan, Euclidean, Chebyshev and Mahalanobis metrics (given dependent dimensions) in 2D space.*

The Canberra metric (for non-negative features).

$$d_{ca}(x, y) = \sum_{i=1}^{D} \frac{|x_i - y_i|}{x_i + y_i} \tag{2.18}$$

The Nonlinear distance.

$$\mathrm{d_N(x,\ y)} = \begin{cases} 0 \quad if \quad d_e(x,y) < T \\ \\ H \quad if \quad d_e(x,y) \geqslant T \end{cases} \tag{2.19}$$

where T is a threshold and H is a distance. An appropriate choice for H and T for feature selection is that they should satisfy $H = \frac{\Gamma(p/2)}{T^p 2\sqrt{\pi^p}}$ and that T satisfies $T^p N \to \infty, T \to 0$ as $N \to \infty$

The correlation metric

$$Corr(x,y) = \frac{\sum_{i=1}^{D}(x_i - \hat{x})(y_i - \hat{y})}{\sqrt{\sum_{i=1}^{D}(x_i - \hat{x})^2 \sum_{i=1}^{D}(y_i - \hat{y})^2}} \tag{2.20}$$

The Angular metric

$$Ang(x,y) = \frac{\sum_{i=1}^{D} x_i y_i}{\sqrt{\sum_{i=1}^{D} x_i^2 \sum_{i=1}^{D} y_i^2}} \tag{2.21}$$

Note that while the correlation metric and angular metric are similarity measures, the others are measures of dissimilarity.

K-Means Algorithms

Once we have compute a measure of similarity or dissimilarity, a criterion function to be optimized it is needed. The most widely used criterion function for classifying is the sum-of-square-error, which is define as:

$$J_{MSE} = \sum_{i=1}^{C} \sum_{x \in \omega_i} |x - \mu_i|^2 \tag{2.22}$$

$where \mu_i = \sum_{x \in \omega_i} x$

This criterion measures how well the data set $X = x_1, x_2, \ldots, x_N$ is represented by the centroids $\mu = \mu_1, \mu_2, \ldots, \mu_C$ (C<N). The classifying methods that use this criterion are called *minimum variance methods.*

Other criterion functions exist, based on the scatter matrices used in Linear Discriminant Analysis (LDA) as it is explained in [5]

The k-means algorithm is a simple classifying procedure that attempts to minimize the criterion function J_{MSE} iteratively:

1. Define the number of classes.
2. Initialize classes by an arbitrary assignment of examples to classes or an arbitrary set of centroids (some examples used as centers).
3. Compute the sample mean of each class.
4. Reassign each example to the class with the nearest mean.
5. If the classification of all samples has not changed, stop, else go to the first step.

Hierarchical classifying methods

Hierarchical clustering methods (in which the clusters can be represented into a Dendrogram, shown in figure 2.10) can be grouped in two general classes

1. Agglomerative (also known as bottom-up or merging): Starting with N single clusters, successively merge clusters until one is left. We can sum this algorithm in the following steps (being N_C the number of cluster and N_{EX} the number of examples):

   -Start with one large cluster.

   -Find "worst" cluster. (Largest number of examples, largest variance, largest sum-squared-error..)

   -Split it (Mean-median in one feature direction, perpendicular to the direction of largest variance..)

   -If $N_C < N_{EX}$ go to 1.

2. Divisive (also known as top-down or splitting): Starting with a unique cluster, successively split the clusters until N single examples are left. The steps of this algorithm are:

   -Start with N_{EX} singleton clusters.

   -Find nearest clusters (with the minimum, maximum, average or mean distance).

   -Merge them.

   -If If $N_C > 1$ go to 1

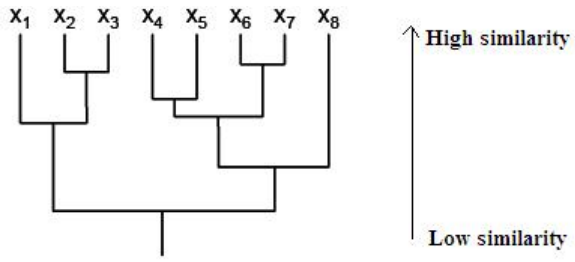


Figure 2.10: Dendrogram. *That is the representation used for hierarchical clusters. It consists on a binary tree that shows the structure of the clusters. In addition to the binary tree, the dendrogram provides the similarity measure between classes(the vertical axis).*

2.2.2 Support Vector Machines

In this section, it is briefly explained the principles of SVM, a more detailed explanation will be found in the chapter 3.

SVM in its simplest form, linear and separable case, is defined as the optimal hyperplane that separates the vector sets belonging to different classes with the maximum distance to its closest samples, called support vectors. The problem is solved using a particular Lagrange formulation in which the problem is reduced to the computation of Lagrange multipliers.

SVM in its general form, non-linear and non-separable, is very similar to its simplest form. Non-separable cases are considered by adding an upper bound to the Lagrange multipliers, and non-linear cases are considered by replacing all the inner products like $(\cdot,\cdot)$, by a so-called kernel function $K(\Delta,\Delta)$. Thus, the system to be solved corresponds to the following 2.23 and it is completely described in 3.5.2.

$$argmax L_D(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(,) \tag{2.23}$$

2.2.3 Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of it is the novel structure of the information processing system: it is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems.

An important application of neural networks is pattern recognition (as it is explained in [6]): it can be implemented by using a feed-forward (figure 2.11) neural network that has been trained previously. During the training, the network is trained to associate outputs with input patterns. When the network is used, it identifies the input pattern and tries to output the associated output pattern. The power of neural networks comes to life when a pattern that has no output associated with it, is given as an input. In this case, the network gives the output that corresponds to a taught input pattern that is least different from the given pattern.

There are two possibilities in the architecture of neural networks:

The feed-forward networks which allow signals to travel one way only;

from input to output. There is no feedback (loops) that means that the output of any layer does not affect that same layer. Feed-forward Networks tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top-down.

The feedback networks that can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations.

*Network layers.*The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units. The activity of the input units represents the raw information that is fed into the network, the activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units and the behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

Perceptrons. The perceptron turns out to be an MCP model (neuron with weighted inputs) with some additional, fixed, preprocessing.

Multilayer Perceptrons. MLPs are feed-forward networks of simple processing units with at least one hidden layer. Each processing unit operates in a similar way as the perceptron, except for the threshold function is replaced by a differentiable non-linearity.

A differentiable non-linearity is required to ensure that the gradient can

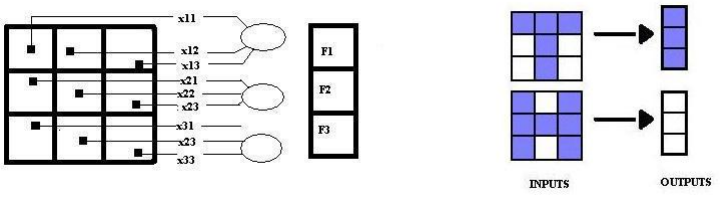


Figure 2.11: *Feed-forward Neural Network. This network is trained to recognise the patterns T and H. The associated patterns are all black and all white respectively.*

be computed. The critical feature in MLPs is the non-linearity at the hidden layer. If all neurons in an MLP had a linear activation function, the MLP could be replaced by a single layer of perceptrons, which can only solve linearly separable problems.

The MLP learning problem is that of finding the weights W that capture the input/output mapping implicit in a dataset of examples. To do it, it is necessary following a complex algorithm which is described in detail in [7]

Chapter 3

Face Verification based on SVM.

We want to design an algorithm able to authenticate a person given an image of him, which means deciding if that person is the supposed client or he is an impostor, and use the technique of SVM in the classification step.

As it has been explained before, the first step in an authentification process is to reduce the dimensionality of the face representation space and then, the classification takes place.

In this chapter it is explained in detail the methods we have tested in order to decide which is, in our opinion, the best to design such a face verification algorithm. First, we will start with an introduction to the problem, and then, we will explain the techniques (figure 3) tested to perform the dimensionality reduction as well as the ones used to the posterior classification.

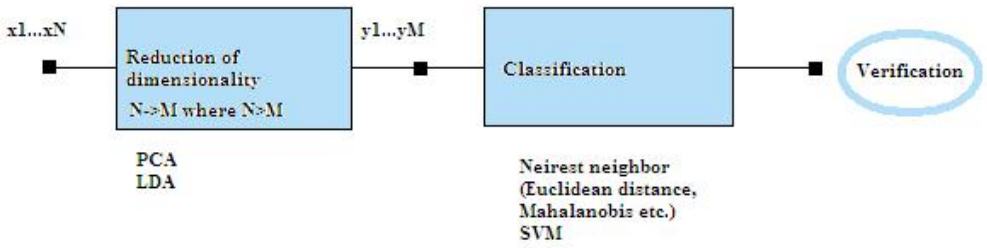


Figure 3.1: *Techniques used in the process of verification*

3.1 Introduction to the problem

As it has been explained before, the problem of face recognition/verification is a challenging work that has attracted scientists from several fields.

The main idea of verification is very simple, but the problems that appear during the process are difficult to solve. The main problems that we have to manage during the process of verification, as it has said before, are the deformability of a face, and the lighting variation. So that, in this chapter we will especially consider this two problems and we will present some strategies that have been proposed in order to overcome them.

3.2 PCA.

PCA is a feature extraction technique commonly used for dimensionality reduction.The objective of it is to perform dimensionality reduction while preserving as much of the randomness (variance) in the high-dimensional space as possible.

3.2.1 The algorithm

In this section we will describe PCA in detail from the mathematical point of view.

The objective of this algorithm is to find the components y that they explain the maximum amount of variance possible by m-linearly transformed components. PCA can be defined in an intuitive way using a recursive formulation.

$$y = W^T x$$

- For PCA to work properly, it is necessary to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension. So, all the x values have the $\overline{x}$ (the mean of the x values of all the data samples) subtracted. This produces a data set whose mean is zero.

  So that, in order to compute the mean, it is necessary to compute the mean of the images in order to center the database, and for it we used the following

$$\overline{X} = \frac{1}{N} \sum_{i=1}^{N} X_i \tag{3.1}$$

And then, the mean is removed from all the images:

$$\widetilde{X_i} = X_i - \overline{X}, i = 1, ..., M. \tag{3.2}$$

All the vectors are ordered in a matrix, that we call A,

$$A = [\widetilde{X}_1 \widetilde{X}_2 ... \widetilde{X}_M] \tag{3.3}$$

- Besides, the Scatter matrix is calculated using the following definition

$$S_T = AA^T \tag{3.4}$$

so that, the scatter of the transformed feature vectors $[y_1, y_2, ..., y_N]$ is

$$W^T S_T W \tag{3.5}$$

.

- The next step is where we find the projection. In PCA, the criterion to choose the projection W_{opt} is maximizing the determinant of the total Scatter Matrix of the projected samples. We can express it mathematically as

$$W_{opt} = \arg\max_W \left| W^T S_T W \right| = [w_1 w_2 ... w_M] \tag{3.6}$$

where $[w_1 w_2 ... w_M]$ is the set of n-dimensional eigenvectors of S_T which are ordered from the largest to the smallest corresponding eigenvalues.

- The latest step in the algorithm consist on deciding the dimension of the subspace where we want to project the samples. The number m of eigenvectors corresponding to the largest m eigenvalues are selected to construct the orthogonal base of the face representation space, where we will project all the samples to reduce the dimension.

  The value of m is selected depending on the total variation of the data which is shown in figure 3.2: It is usual to choose the number of eigenvalues that keeps 90% of the total variation.

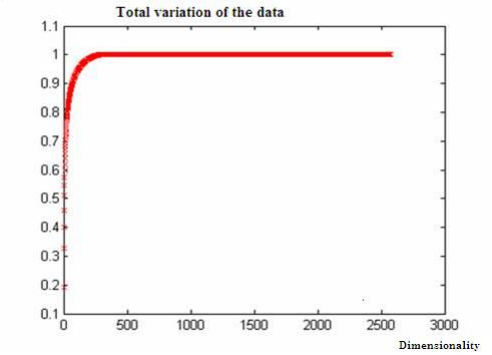


Figure 3.2: *Total Variation of the data*

3.2.2 PCA: Lighting Variation and Face Deformability

As we have said before, lighting variation and face deformability are two of the main problems that are necessary to overcome in face verification, because it can strongly affect in the results.

The more little our training database is, the more critical these problems are, specially in the case of lighting variation, because if our database is large enough it is possible to model the lighting variation.

Therefore, it is necessary to give a small sum up of what it is said in the literature about this effect and the possible solutions the experts propose.

Lighting Variation

It is suggested in [8] that we can face the problems derived to lighting variation by discarding the first three eigenvectors that we obtain: we order the eigenvectors starting with the one belonging to the biggest eigenvalue and we take the m eigenvalues starting from the fourth one.

This was suggested because it is affirmed in [8] that those eigenvector are the ones which capture the variation due to the lighting. So that, if we do not consider that information, there will be less errors derived to the strong differences between the pictures of the training set and the pictures of the test set due to the lighting variation.

According to [8], several experiments have shown that this algorithm performs better if the lighting is nearly frontal in the picture.

Variation in facial expression

About variation in the facial expressions, it is said in [8] that for PCA the more eigenvectors we take, the better the performance is.

Anyway, in the cited paper, this algorithm is presented as the worst compared to Fisherfaces, Correlation and Linear subspace in presence of facial variations if we want to recognize/verify the whole face. Moreover, if we apply this algorithm to recognize if someone wears glasses the results obtained are really bad.

3.2.3 Advantages and limitations of PCA

PCA has three well-known characteristics, that have made it one of the most used algorithms:

- *Non-iterative global optimization of a natural cost function* That implies more mathematical simplicity, which it brings us to the next advantage: better computational efficiency.
- *Computational efficiency.* When we get the matrix defining the subspace were we will project the samples, the reduction of the face dimension is easy to perform: It just consists in a simple projection.

But PCA also presents some limitations that it is necessary to take into account:

- Since PCA uses the eigenvectors of the covariance matrix S_T, it is able to find the independent axes of the data under the unimodal Gaussian assumption, but in the non-Gaussian or multimodal case, PCA simply de-correlates the axes.
- PCA does not consider class separability since it does not take into account the class label of the feature vector. It simply performs a coordinate rotation that aligns the transformed axes with the directions of maximum variance. Besides, there is no guarantee that the directions of maximum variance will contain good features for discrimination.

3.3 LDA

The objective of LDA is to perform dimensionality reduction while preserving as much of the class discriminatory information as possible.

Since, normally, the learning set is labelled, it makes sense to use this information to build another algorithm that successfully reduces the dimensionality of the feature space.

Fisher's Linear Discriminant (FLD) is an example of a class specific method that selects W in such a way that the ratio of the between-class scatter and the within-class scatter is maximized.

3.3.1 The algorithm

In this section the algorithm used to compute the optimal W is explained:

Defining between-class scatter matrix as

$$S_B = \sum_{i=1}^{C} N_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{3.7}$$

and the within-class scatter matrix as

$$S_W = \sum_{i=1}^{C} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T \tag{3.8}$$

where μ_i is the mean image of the class X_i.

If S_W is nonsingular, the optimal projection W_{opt} is chosen as the matrix with orthonormal columns which maximizes the ratio of the determinant of the between-class scatter matrix of the projected samples to the determinant of the within-class scatter matrix of the projected samples:

$$W_{opt} = \arg\max_W \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|} = [w_1 w_2 ... w_M] \tag{3.9}$$

where $(w_i|_{i=1,2,...M})$ is the set of generalized eigenvectors of S_B and S_W corresponding to the M largest generalized eigenvalues λ_i $S_B w_i = \lambda_i S_w w_i$.

3.3.2 LDA: Lighting Variation and Face deformability

Lighting Variation

In the literature [8] is said that this algorithm performs better when lighting is frontal, and when the angle of the lighting variation is not frontal it

performs better than PCA method.

In general it is said that the Fisherfaces method, performs better under lighting variations than PCA method, even when the first three eigenvectors have been removed.

Variation in Facial Expression

Since the Fisherface method tends to discard the those parts of the image which ate not significant for recognizing, the resulting projections of W do not include so much information of those parts of the face that presents a high variation. That characteristic of LDA makes it performs better than other techniques such as PCA, Linear Subspace and Correlation methods under different conditions of facial expressions.

3.3.3 Advantages and Limitations of LDA

Using LDA to reduce the dimension have some advantages related to the classification: since it maximizes the discriminatory information that should facilitate the posterior classification. if we take one set of samples from two different classes and which randomly lie in a perpendicular direction to a linear subspace and we apply PCA and LDA to reduce their dimension from 2D to 1D then, we can observe (figure 3.3.3) that the classes appear separate in the LDA case, while in the PCA case they appear mixed all together. That means that LDA performs a better between class scatter, so that, the posterior classification should be easier in the LDA case than in the PCA case.

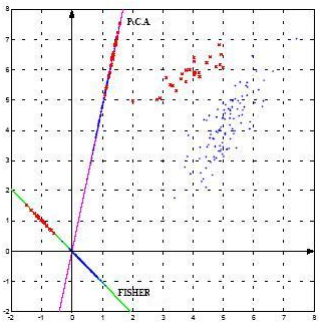


Figure 3.3: *The posterior classification should be easier if we apply LDA than in the PCA case*

Although LDA seems to be a good method to reduce dimensionality, it

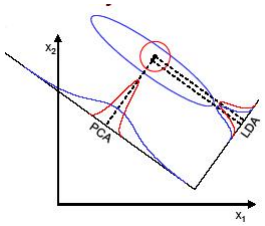
presents some limitations.

- *LDA produces at most C-1 feature projections.*

  There are at most c-1 nonzero generalized eigenvalues, so that the maximum dimensionality should be c-1, where c is the number of classes. If the classification error estimates establish that more features are needed, some other method must be employed to provide those additional features.

- *LDA is a parametric method since it assumes unimodal Gaussian likelihoods.* If the distributions are significantly non-Gaussian, the LDA projections will not be able to preserve any complex structure of the data, which may be needed for classification.

- *LDA will fail when the discriminatory information is not in the mean but rather in the variance of the data.*



3.4 Distance Classification Method

The most simple strategy used in the classification step consists on deciding whether a sample corresponds to a specific centroid depending on the distance between both sample and centroid. The strategy is simple : comparing the distance of the sample projection and the centroid the client is accepted if the distance is below a threshold, otherwise he is rejected.

Since our objective is to perform a verification method, it is equally necessary to define the maximum distance that we consider to decide that one sample corresponds to a specific centroid.

The different metrics we have used to compute the distances are

- Euclidean distance. From a geometrical point of view this distance measures the difference between the reduced face vectors and the difference between the reconstruction error of each vector, given by the difference of the norms.

$$\|x - y\|_e = (\sum_{i=1}^{D} |x_i - y_i|^2)^{1/2} \tag{3.10}$$

- Mahalanobis distance. From a geometrical point of view this distance, as a different metric system, has a scaling effect in the image space: directions in which a greater variance exist are compressed and directions in which a smaller variance exist are expanded.

$$d(x, y) = ((x - y)^T C^{-1}(x - y))^{1/2} \tag{3.11}$$

It can be proved that in the PCA space the Mahalanobis distance is equivalent to the Euclidean distance. In 3.11 we suppose x and y vectors in the image space, and $C^{-1} = W_{PCA}^T \Gamma^{-1} W_{PCA}$. So that, we can rewrite Mahalanobis formula as

$$d(x, y) = ((x - y)^T W_{PCA}^T \Gamma^{-1} W_{PCA}(x - y))^{1/2} \tag{3.12}$$

If we compare this with 3.10 we can see that those distance are equivalent if we weight each component by the inverse correspondent eigenvalues or correspondent variance. This procedure is equivalent to change W_{PCA} to $W_{PCA}^T \Gamma^{-1}$ (called whitening PCA [9]) and then applying Euclidean distance.

The result, after applying such a transformation is that we obtain a non-orthogonal set of vectors in $WPCA$ due to the effect of stretches and shrinks of the axes.

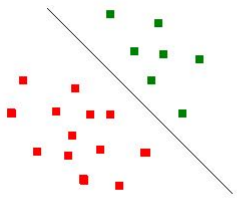
The Mahalanobis metric could be better than the Euclidean metric for the recognition/verification matching problem if each class is normal distributed.

These both classifier were performed to compare the different techniques of dimensionality reduction, and in the same time compare the results obtained with the ones from multiclass-SVM.

3.5 Support Vector Machines.

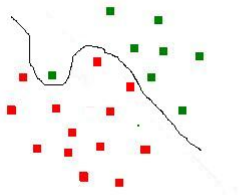
3.5.1 Concept of Support Vector Machines

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. A schematic example is shown in figure 3.5.1. In this example, the objects belong either to class *green or red*. The separating line defines a boundary on the right side of which all objects are *green* and to the left of which all objects are *red*. Any new object (white circle) falling to the right is classified as *green* (or classified as *red* if it falls to the left of the separating line).

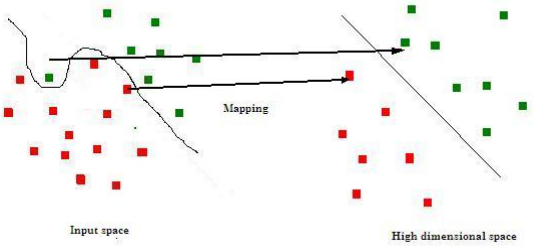


The above is a classic example of a linear classifier: a classifier that separates a set of objects into their respective groups (*green and red* in this case) with a line. Most classification tasks, however, are not that simple, and often more complex structures are needed in order to make an optimal separation, i.e. correctly classify the test samples in base of the samples in a training set. This situation is shown in the picture 3.5.1 Compared to the previous schematic, it is clear that a full separation of the *green and red* objects would require a curve (which is more complex than a line).

Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyperplane classifiers. Support Vector Machines are particularly suited to handle such tasks.



The figure 3.5.1 shows the basic idea behind Support Vector Machines. Here we see the original objects (left side of the schematic) mapped, using a set of mathematical functions, known as kernels. In this new setting, the mapped objects (right) are linearly separable and, thus, instead of constructing the complex curve (left), all we have to do is to find an optimal line that can separate the *green and red* objects.



Support Vector Machine (SVM) is primarily a classier method that performs classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. SVM supports both regression and classification tasks and can handle multiple continuous and categorical variables.

3.5.2 Mathematical background

Introduction

As we have seen below, the stated problem is, given a dataset, learn a function that will correctly classify unseen examples. Such function is found

by optimizing some measure of performance of the learned model. The problem, then, is getting a good measure of performance, and this is the expected risk, defined as follows

$$R[f] = \int C(f(x), y) dP(x, y) \tag{3.13}$$

where $C(f, y)$ is a suitable cost function, e.g the squared error $C(f, y) = (f(x) - y)^2$. Unfortunately, this risk can not be measured, because the pdf is unknown, we can only estimate it using the error made on the known examples (the empirical risk):

$$R_{emp}[f] = \frac{1}{N} \sum_{i=1}^{N} C(f(x_i), y_i) \tag{3.14}$$

Therefore, the next step is to perform the minimization of the empirical risk. For this, the concept of what we have to do is easy: we have to find a function $f(x)$ that minimizes the average risk on the training set. In this process, we should know that the more samples we have in the training set the more the empirical risk will asymptotically converge to the expected risk, that statement is assure by the law of large numbers.

But we can have two different problems: the first problem, the so-called underfitting, appears if we do not have enough data to estimate *Remp*. The second, called overfitting, appear when we use a complicated decision boundary, then we can always have $R_{emp} = 0$ but in this case $\pi[f] -> \inf$. That is the reason why we restrict the class of functions to simple functions ($\pi[f]$ small), so that $R[f] \leqslant R_{emp} + \pi[f]$.

The VC dimension

The Vapnik-Chervonensis dimension is a measure of the complexity (or capacity) of a class of functions $f(\alpha)$, more concretely it measures the largest number of examples that can be shattered by the family $f(\alpha)$.

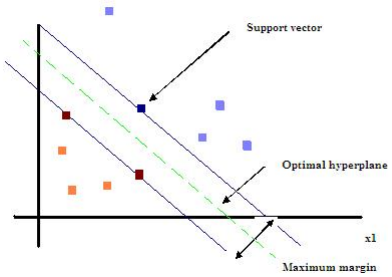
But the criterion is not that simple, in this process we should have a compromise between the high capacity and the ability of generalize. That means that if the family $f(\alpha)$ has enough capacity to classify every possible dataset, we should not expect these functions to generalize very well, and, on the other hand, if functions $f(\alpha)$ have small capacity but they are able to classify a particular dataset it will likely work well also for unseen data.

VC-dimension is relevant because it provides bounds on the expected risk as a function of empirical risk and the number of available examples.

Unfortunately, sometimes it is not practical computing an upper bound using VC dimension, the reason, for example, we can cite some situations : when VC dimension cannot be accurately estimated for non-linear models such as neural networks or when the VC dimension may be infinite.

Optimal separating hyperplanes: separable case

Once we have defined that concepts of empirical risk and VC dimension, we will focus on the problem in which is based the concept of Support Vector Machines: finding an optimal separating hyperplane for a linearly separable dataset. We consider the optimal separating hyperplane as the one with the largest margin (minimum distance of an example to the decision surface). We can see this idea figure 3.5.2.



A VC dimension of a separating hyperplane of a margin m is bounded as follows

$$h \leqslant \min(\left[\frac{R^2}{m^2}\right], D) + 1 \tag{3.15}$$

where D is the dimensionality of the input space and R is the radius of the smallest sphere containing the input vectors.

Therefore, by maximizing the margin we are in fact minimizing the VC dimension, and, since the separating hyperplane has zero empirical error (it correctly separates all the training examples), maximizing the margin will also minimize the upper bound on the expected risk.

Since we want to maximize the margin, let's express it as a function of the weight vector and bias of the separating hyperplane

$$\frac{|w^T x + b|}{\|w\|}$$

Noticing that the optimal hyperplane has infinite solutions by simply scaling the weight vector and bias, we choose the solution for which the discriminant

function becomes one for the training samples closest to the boundary

$$\left|w^T x_i + b\right| = 1$$

Therefore, the distance from the closest sample to the boundary is

$$\frac{\left|w^T x + b\right|}{\|w\|} = \frac{1}{\|w\|}$$

and the margin is

$$m = \frac{2}{\|w\|}$$

Therefore, the problem of maximizing the margin is equivalent to minimize

$$J(w) = \frac{1}{2}\|w\|^2$$

subject to $y_i(w^T x_i + b) \geqslant 1$

To solve this problem, we will use classical Lagrangian optimization techniques. To solve the minimization explained below we introduce the Lagrangian

$$L_P(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(w^T x_i + b) - 1]$$

in which we minimize L_P with respect to the variables w and b and we maximize L_P with respect to the dual variables $\alpha_i \geqslant 0$ (Lagrange multipliers). This is called the Lagrangian primal problem.

If we simplify the primal problem using the Kuhn-Tucker condition we obtain the Lagrangian dual problem

$$L_D(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

The primal problem scales with dimensionality (w has one coefficient for each dimension), whereas the dual problem scales with the amount of training data (one Lagrange multiplier per sample). The Karush-Kuhn-Tucker complementary (described in Annex B) condition states that, for every point in the training set, the following equality must hold

$$\alpha_i [y_i(w^T x_i + b) - 1] = 0$$

with $\forall i = 1...N$. Those points for which $\alpha_i > 0$ must lie on one of the two hyperplanes that define the largest margin. These points are known

as *Support Vectors*. All the other points must have $\alpha_i = 0$ Once we have defined the support vectors, we can replace the complete database by those points, and the separating hyperplane will be the same.

We can also give an expression of the separating hyperplane as function of the support vectors

$$h_{\alpha_i,b}(x) = \sum_{i,\alpha_i>0} y_i \alpha_i \langle x_i, x \rangle + b \tag{3.16}$$

The non-separable case

The last case is used in problems that were linearly separable, but the theory can be generalised to the case of datasets which are not linearly separable. The solution to it consists on introducing slack variables ξ_i that relax the constraints of the canonical hyperplane equation

$$y_i(w^T x_i + b) \geqslant 1 - \xi_i$$

$\forall i = 1...N$

The objective of introducing that slack variables is to measure deviation from the ideal condition. When having that slack variables, the optimization problem change a little bit: now the goal will be to find a hyperplane with minimum misclassification rate. To accomplish it the function to minimize will be

$$\Phi(\xi) = \sum_{i=1}^{N} I(\xi_i - 1)$$

that represents the total number of misclassifications. It can be approximated by

$$\Phi'(\xi) = \sum_{i=1}^{N} \xi_i$$

which is an upper bound on the number of misclassifications, and minimize the joint objective function

$$\Phi(\xi) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i$$

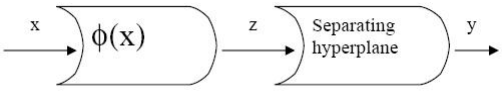
subject to $y_i(w^T x_i + b) \geqslant 1 - \xi_i$ and $\xi_i \geqslant 0$. C represents the compromise between capacity and misclassification (large values of C will imply few misclassification errors and small values will imply low complexity solutions)

The process to obtain the support vectors is the same as in the separable case, using the Lagrangian dual problem.

In the next sections we will present the different models for classification that derive from this mathematical theory.

Non-linear SVMs

According to the Cover's theorem ("A complex pattern-classification problem cast in a high-dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space ") SVMs powerful resides in the fact that in order to perform the classification, they first do a non-linear mapping of the feature vector onto a high-dimensional space and then they construct a separating hyperplane in that high-dimensional space.



This is performed by using the Kernels, which means replacing $< \Delta, \Delta >$ by a kernel function in the equation 3.16. The kernel functions that we can choose are:

-Linear: $K(x_i, y_i) = x_i^T x_j$.

-Polynomial: $K(x_i, y_i) = (\gamma x_i^T x_j + \tau)^d, \gamma > 0$.

-Radial Basis Function(RBF): $K(x_i, y_i) = exp(-\gamma \|x_i^T x_j\|^2), \gamma > 0$.

-Sigmoid: $K(x_i, y_i) = tanh(\gamma x_i^T x_j + \tau), \gamma > 0$.

where,γ, τ and d are kernel parameters.

3.5.3 SVM Models

To construct an optimal hyperplane, SVM applies an iterative training algorithm, which is used to minimize an error function. According to the form of the error function, SVM models can be classified into four different groups:

- Classification SVM Type 1 (also known as C-SVM classification)
- Classification SVM Type 2 (also known as nu-SVM classification)
- Regression SVM Type 1 (also known as epsilon-SVM regression)
- Regression SVM Type 2 (also known as nu-SVM regression)

C-SVM Classification

For this type of SVM, training involves the minimization of the error function

$$\frac{1}{2}w^T w + C\sum_{i=1}^{N}\xi_i \tag{3.17}$$

subject to the constraints $y_i(w^T\phi(x_i)+b) \geqslant 1-\xi_i$ and $\xi_i \geqslant 0$, $i = 1, ..., N$

where C is the capacity constant, w is the vector of coefficients, b a constant and ξ_i are parameters for handling nonseparable data (inputs). The index i labels the N training cases. Note that $y = -1, +1$ is the class labels and x_i is the independent variables. The kernel is used to transform data from the input (independent) to the feature space. It should be noted that the larger the C, the more the error is penalized. Therefore, C should be chosen with care to avoid over fitting.

nu-SVM Classification

In contrast to Classification SVM Type 1, the Classification SVM Type 2 model minimizes the error function

$$\frac{1}{2}w^T w - \upsilon\rho + \frac{1}{N}\sum_{i=1}^{N}\xi_i \tag{3.18}$$

subject to the constraints $y_i(w^T\phi(x_i) + b) \geqslant \rho - \xi_i$ and $\rho \geqslant 0$, $\xi_i \geqslant 0$, $i = 1, ..., N$.

3.6 Multi-class SVM

SVM is a method used to classify the samples within two classes but, since our goal is to perform a face verification method, which implies normally having more than two classes (identities, in our case) it is necessary to extend the theory from a 2-class classifier to a c-class classifier.

To do it there are several techniques we can use:

- *One against the others (1vsAll)*

  This method is also known as winner-take-all classification. For an M class classification, M binary SVM classifiers are created. Each classifier is trained to discriminate one class from the remaining M-1 classes. During the testing step, data are classified by computing the margin from the optimal separating hyperplane.

- *Pairwise classification method or One against one (1vs1)*

  In this method, SVM classifiers for all possible pairs of classes are created. For an M class classification, we create $M(M-1)/2$ binary classifiers. Each binary classifier is trained to classify two classes of interest. During the testing or application phase, the output from each binary classifier in the form of a class label is obtained. The class label that occurs the most is assigned to that data. We adopt a tie-breaking strategy in case of a tie. A common tiebreaking strategy is to randomly select one of the class labels that are tied.

- *Classification based on Directed Acyclic Graph (DAG)*

  This method is based on the Decision Directed Acyclic Graph structure that has a tree-like structure. Similar to the pairwise classification method, we create $M(M-1)/2$ binary classifiers for an M class classification. Each binary classifier is trained to distinguish two classes and forms a node in the graph structure. Nodes are organized in the form of a triangle with the single root node at the top and increasing subsequently in an increment of one node in each level until the last level that will have M nodes (see figure **??**. The DAG evaluates an input data starting at the root node (top node) and moves to the next level based on the output values. The binary classifier in the next level then evaluates the input data. The path traversed by data is called the evaluation path. The DAG method eliminates one class out from the list at each level. At the root node, all classes are in the list. Each node discriminates between the first class and the last class in the list. Each level gives the result in one class out of the two classes; the class that is not in favor of that level is eliminated from the list. The procedure is terminated when only one class remains in the list. Although,

here the number of binary classifiers equals the number of classifiers required by the pairwise classification method, inputs are evaluated only $M - 1$ times resulting in faster classification.

In order to perform the classification of M classes (in our case M=52) we will use the first two techniques exposed. In the case of the technique 1vsAll, we have two options to train the model:

- Given C identities, to train the model for identity n we use some samples from the identity n as samples type +1 and the samples from the rest of identities as samples type -1.
- Given C identities, we train identity n by using samples from that identity as samples type +1 and all the samples from a generic database "WM" as samples type -1.

The second strategy presents the advantage that if we want to extend the number of identities (classes) of the system, we will not have to re-train all the models of the identities, we will just use the generic database WM and samples from the new identity to train the model of that new class. That is the reason why we will try the second strategy in order to extend the SVM 2-class method to a multi-class method. co

Chapter 4

Results

In this chapter we present the results obtained with the face verification algorithm created, as well as a little introduction where we expose the characteristics of the database used (BANCA), the protocol associated to this database and the measures taken during the application of such a protocol in order to "quantify" the performance.

Finally, we will compare the results obtained with the ones obtained using other technologies.

4.1 BANCA

BANCA is a European Project whose aim is to develop and implement a secure system with enhanced identification, authentification and an access control schemes for applications over the Internet. One if the innovations of this project is that it improves the security system by combining the classical security protocols with robust multi-modal verification schemes based on speech and face images.

When building a recognition or verification system, it is necessary to have a large data set for the training step, and generally, the larger the training set is, the better the performance achieve. For multi-modal systems, the necessary training data set is in the order of TBytes, that means an extraordinary computational capacity which was unreachable some years ago, however, nowadays it is possible to work with such a large databases and manipulate it to use it effectively.

For the BANCA protocol it was necessary to get a multi-modal database that contained a wide range of realistic recording scenarios,so that to build

that database it was necessary to use a variety of materials and different European languages.

There are three other publicly available medium or large scale multimodal databases to evaluate verification and recognition algorithms, such as M2VTS which comprise 37 subjects, DAVID-BT and the database collected within the Extended M2VTS EU project. Finally, there are some other databases but they are mono-modal, such as FERET,Yale, Hardvard or Olivetti.

4.1.1 Specification of BANCA Database

BANCA database was designed to test multi-modal IV with different acquisition devices (2 cameras and 2 microphones) and under several scenarios (controlled, degraded and adverse).It comprises the video and speech data of 52 clients (26 males and 26 females) collected on 12 different occasions, and this was done for four different languages (English, French, Italian and Spanish), that means a total of 208 people. Each language and gender specific population is subdivided in two groups (g1 and g2) of 13 subjects each.

Each subject recorded 12 sessions, each of them containing 2 recordings: one true client access and one impostor attack (where the client was previously inform about what the claimed identity was suppose to utter). Those 12 sessions where divided into three different scenarios, as we have explained before: *Controlled*(sessions 1-4),*Degraded*(sessions 5-8) and *Adverse*(sessions 9-12).

4.1.2 Experimental protocol

In verification two types of protocols exist, closed-set and open-set. In the close-set verification the population if clients is fixed, this means that the system design can be tuned to the clients in the set.Thus both, the adopted representation and verification algorithm applied in the feature space are based on some training data collected for this set of clients. Anyone who is not in the training set is considered as an impostor. In open-set verification we wish to add new clients to the list without having to redesign the verification system. In particular we want to use the same feature space and the same design parameters such as thresholds.In such a scenario the feature space and the verification system parameters must be trained using completely independent data from that used for specifying client models. BANCA is an example of an open-set verification protocol.

Session	MC	MD	MA	UD	UA	P	G
1	TT			TT	TT	TT	TT
2	T					T	T
3	T					T	T
4	T					T	T
5		TT					TT
6		T		T		T	T
7		T		T		T	T
8		T		T		T	T
9			TT				TT
10			T		T	T	T
11			T		T	T	T
12			T		T	T	T

Table 4.1: *BANCA Protocol*

From another point of view, although BANCA is multi-modal, the interest of this project is just about image and that is the reason why we will use the Monolingual Protocol. That means that we only have taken the pictures corresponded to one of the languages.

Monolingual Protocol

First, to define an experimental protocol, it is necessary to define an evaluation set and decide which of the images of its content we will use to train (training set) and which ones we will use to test(test set).

The BANCA Protocol specify seven different configurations depending on which sessions are used for training and which for testing. These seven configurations are Matched controlled (MC), Matched degraded (MD), Matched adverse (MA), Unmatched degraded (UD), Unmatched Adverse(UA), Pooled test (P) and Grand test (G). Table 4.1 show which sessions are taken for the training set and which for the test set in each of the different configurations.

For each configuration, in table 4.1 the sessions used for training are labelled TT and the ones used for testing are labelled T. So, for the client testing we used the sessions marked with a T and for the impostor testing we used the TT and the T ones.

After applying the BANCA protocol we can compute several results: the

intrinsic performance in a given condition, the degradation from a mismatch between controlled training and uncontrolled test,the performance in varied conditions with only one (controlled) training session, or the potential gain that ca be expected from more representative training conditions.

4.2 Measures

In order to visualize the performance of our system, we have used several kind of measures:

- *False acceptance rate* (FAR): Is the ratio between the number of false acceptance and the number of impostor accesses.

- *False rejection rate* (FRR): Is the ratio between the number of false rejections and the number of impostor accesses.

This last two types of error have an associated cost which is denoted as CFA and CFR.

- *DET curve* which plots on a log deviate scale the FRR as the function of the FAR. The point where FRR = FAR is called Equal Error Rate (EER).

We also can compute the performance of our system under 3 different conditions corresponding to three different values of the cost ratio $R = \frac{CFA}{CFR}$, with different values of R:

R= 0.1 FA is an order of magnitude less harmful than FRR.

R= 1 FR and FA are equally harmful.

R=10 FA is an order of magnitude more harmful than FR.

Once we fix R, we can compute the Weighted Error Rate (WER) which is defined as:

$$WER(R) = \frac{FRR + RFAR}{1 + R} \tag{4.1}$$

We should point out that depending on the value of the decision threshold,θ FRR and FAR can strongly vary (and, consequently, also WER (**??**). So that, it is more efficient to take the θ that minimizes the WER on the training set. This threshold is called *a priori threshold*

$$\theta_{\mathrm{R}} = \mathrm{argmin}_{\theta \mathrm{R}}\, WER$$

Other option is to obtain the θ minimizing the WER but on the testing set. This threshold is called *a posteriori threshold*. Obviously, it is more efficient the *a posteriori threshold* than the *a priori threshold* but it does not correspond to a real situation, because, in this case we are using the

information of the same pictures we use to test. Thus, it is also interesting to compute a plot where it is shown the strong dependency between WER and θ_R.

4.3 Experiments and Results

In this section we present the experiments done while testing our algorithm as well as the results we got.

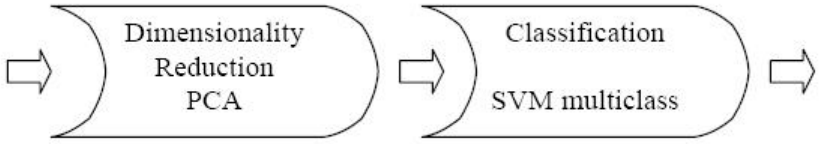


Figure 4.1: *Techniques used to perform the verification*

As we can see in the scheme shown in figure 4.1, there are basically two steps in the process: Dimensionality reduction and classification. So that, in the first part of this section we will present the experiments and the results obtained testing different techniques of dimensionality reduction and simple classifiers, and, in the second part it is explained the procedure we have follow to construct a multiclass SVM and the final experiments done to test the algorithm as well as its performance.

4.3.1 PCA and Distance-based Classifier

PCA was our first choice to perform the reduction of the face image dimension. There are several reasons for it:

- It is the first and most famous method used to perform the dimensionality reduction.
- Its mathematical simplicity and, consequently, its good computational efficiency.

First approach: Face recognition with ORL Database

The implementation of a face recognition system using PCA and a distance-based classifier (using Euclidean distance) was done in Matlab and in the first tests we did not used the BANCA database, but the ORL. This database is not as large as BANCA (there are just 40 different people and 10 images of each) but it was used at the beginning to have a first approach of the results we can get with PCA.

Although this database was too simple to verify the performance of PCA and a simple distance classifier, we tried to do some experiments on it:

1. We varied the number of eigenvectors chosen to form the base of the face representation space.
2. We avoid the first three eigenvectors when choosing the base of the face representation space (as it is suggested in [8] to face the effects of lighting variation)

And the results we got are shown in the following graphic:

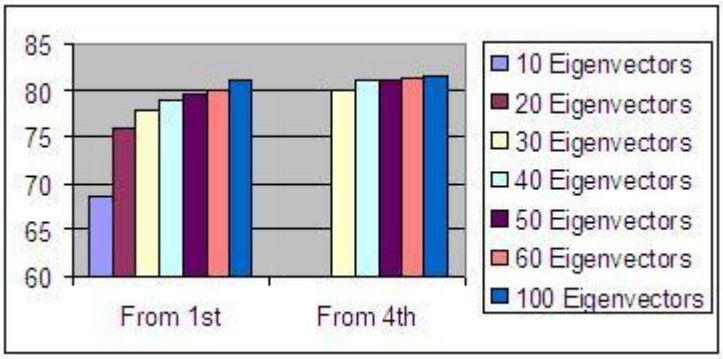


Figure 4.2:

The picture 4.2 show how important is choosing the correct dimension of the face representation space (number of eigenvectors) for recognition, as we can see, the best performance is reached for 100 eigenvectors. But there is not a big difference between taking 60 and taking 100. This is because for 60 eigenvectors we reach almost 99% of the total variation, and the same for 100 eigenvectors.

About the influence of removing the first three eigenvectors, we can see there is an improvement. At first we can think that what it is said in the

cited paper is true, but in the ORL database there is no lighting variation in the pictures but there is variation in the pose or in the facial expressions. However, since the database used is not so large, the reliability of these conclusions is limited.

Application of Banca Protocol

Since the ORL database is too small to apply a Protocol to test the algorithm reliably , it was necessary to use another database specially built for this objective. There are several medium/large databases available in the field of face authentification, some examples are the Yale database, BANCA[1], FERET[10] or M2VTS[11].

In this project, it was established the use of BANCA database in order to test the system, because of its properties (described in detail in 4.1) and its particular protocol associated (described in the same section). Using BANCA[1] will allow us comparing our results with the ones obtained by other groups of work, such as IDIAP [12], in order to have a reference of our archievements.

So, we tested the face recognition system built using PCA and two distance-based classifiers for some scenes described in the BANCA protocol. Since the preliminary experiments showed there is an improvement while removing the first three eigenvectors, in the next experiments we will use the matrix W_{PCA} obtained with this strategy.

	MC	MD
Euclidean distance	82%	71%
Mahalanobis distance	60%	51%

Table 4.2: *Results obtained for Face Recognition to the approach PCA + Euclidean/Mahalanobis distance-based Classifier*

But the goal of this project was building a face verification system, so that we changed the implementation in order to perform the authentification and we completely applied the BANCA protocol taking the measures described in 4.2.

Since we used a distance-based classifier to perform the verification, it was necessary to decide an appropriate threshold (the idea is shown in figure 4.3). During the experiments, we could see the importance of choosing an appropriate threshold. So that, we did some tests to decide its value.

We obtained that for each scene of BANCA there was a different optimal

threshold so we had two options: computing the best threshold for each scene and apply it to have the best performance or taking one threshold value (the mean of the optimal values of all the scenes) for all the scenes. We decided to follow the first strategy.

So that, in table 4.3 we present the results obtained after the application of the BANCA Protocol for PCA + Euclidean distance-based classifier:

(%)	MC	MD	MA	UD	UA	P	G
FRR	28,59	34,65	12,63	71,35	58,09	52,01	49,5
FAR	9,52	25,34	30,10	4,41	9,41	7,06	8,38
WER(R=1)	19,06	29,9	27,69	37,88	33,66	29,53	28,94

Table 4.3: *Results obtained applying BANCA Protocol to the approach PCA + Euclidean distance-based Classifier*

We also tested BANCA for a Mahalanobis distance-based classifier, unfortunately the percentages obtained showed that there was no improvement

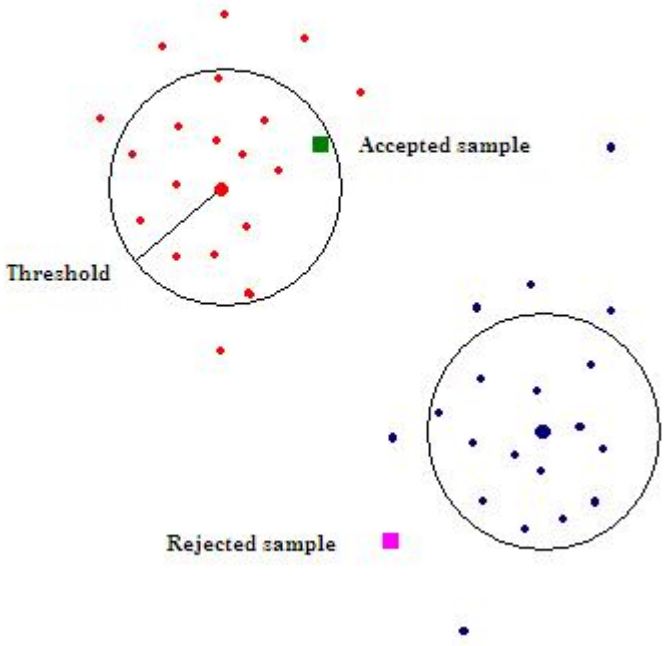


Figure 4.3: *Distance-based verification system*

with respect to the Euclidean case. According to the theory, this results are worst than for the Euclidean case because the classes are not normal-distributed (as it is further explained in section 3.4).

	MC
WER (R=0,1)	8,33
WER (R=1)	27,53
WER (R=10)	7,47

Table 4.4: *MC BANCA Protocol applied to the approach PCA + Mahalanobis distance-based Classifier*

In the graphic 4.4 we present the comparison between the results obtained by IDIAP [12] using a more complex algorithm (LDA for the reduction of dimensionality and SVM for classification) and the results obtained by applying a PCA with a simple Euclidean distance-based classifier and the ones obtained using Mahalanobis distance-based classifier.

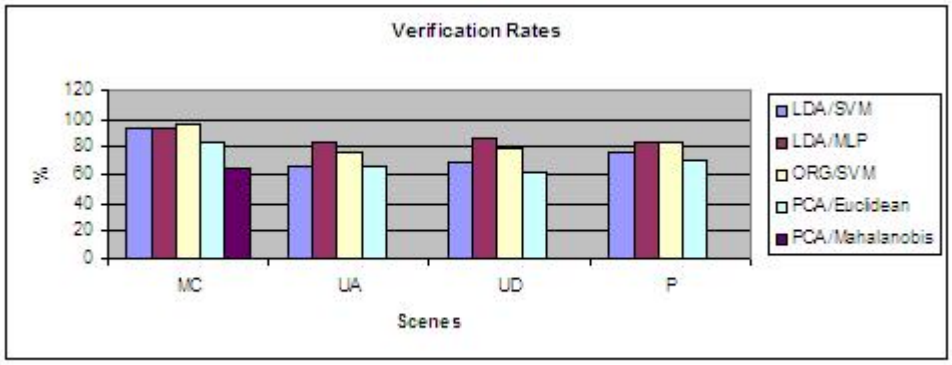


Figure 4.4: *Comparison with other algorithms used in IDIAP*

4.3.2 LDA and Distance-based Classifier

The next experiment was to develop a fisherfaces based system of face verification using a distance-based classifier. So that, it was implemented in matlab and tested using the same classifier as the one used for PCA, with two different options: Mahalanobis distance and Euclidean distance.

Although the results obtained while testing the algorithm with the training set in MC scene were great (98%), the results (near chance) for the test

set showed that there was a problem in the model obtained, so that, we did not applied the rest of the BANCA protocol.

At this point, since the goal of the project was to construct a face verification algorithm with a SVM-based classifier, we decided not to apply LDA to perform the reduction of dimensionality. There were several reasons for taking this decision:

- LDA is better than PCA because it maximizes the inter-class variance and minimize the intra-class variance in the new subspace which facilitates the classification

  But we were going to use an SVM classifier that, according to the theory, first it is done a non-linear mapping of the feature vector onto a high-dimensional space in order to transform the non-linear case to a linear case. So that, the transformation to a linear case was going to be done in the classification step anyway, and the improvements that we could have performed rebuilding all the algorithm were not going to be much better that the ones obtained with PCA.

- The computational cost of LDA is bigger than for PCA. Moreover, it was necessary to optimize the computational cost of the whole system, because it was supposed to be used in a real application which has to be as fast as possible.

4.3.3 PCA and Multiclass SVM classifier

Having got good results for a PCA and Euclidean distance-based classifier, we wanted to improve those results using a Support Vector Machines classifier instead of the distance-based one.

For the classification step we used the library libSVM v-2.7 (see Appendix A) which is described in detail in AnnexA. Figure 4.5 shows how is performed the prediction using libSVM (note there is a cross validation step, where C and γ are decided)

As we have explained in chapter 3 while applying SVM in the classification step there are several variables that it is necessary to fix

- SVM type.
- Kernel type.
- Degree of the kernel.

- Gamma variable of the Kernel function.
- Coef0 in the Kernel function.
- The value of parameter C (cost)
- The epsilon parameter or tolerance of termination criterion.

So, in order to choose the parameters that would optimize our results we did several experiments, but the first one consisted on testing the influence of the scaling step, recommended in [13], paper writen by the authors of the library libSVM-2.7.

Influence of the scaling step

This experiment consisted on evaluating the influence of the scaling step in the performance of the system. This was done in a simple way: we chose some values for the different parameters of the SVM, that can seem a priori appropriate to have good results, and we computed the error rate performing the verification with and without the scaling step. This was done for the scene Matched Control (MC).

But we took another measure that we thought was important: the computational time. This was done because scaling such a big matrix would take a lot of time, and we wanted to study if this step was really necessary or we could avoid it because it implied enlarging the computational cost of the system.

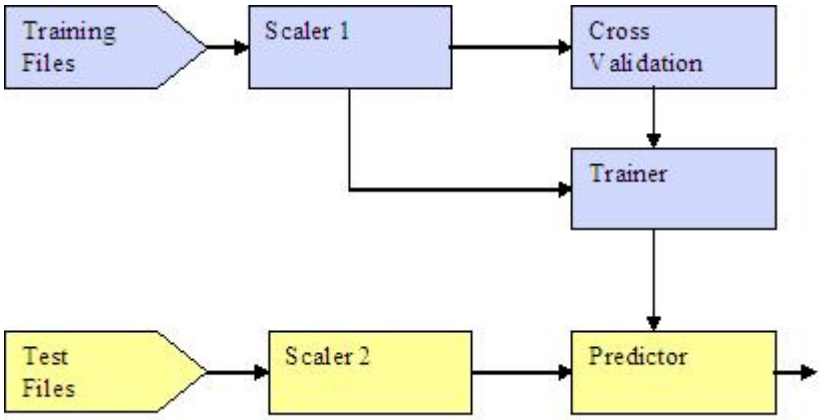


Figure 4.5: *Blocs diagram SVM*

The parameters of the SVM were: C-SVM with 2nd degree polynomial kernel. And the rest of parameters were the default ones in libSVM-2.7

The results were the following:

	No scaling	Scaling
WER	18,0%	15,22 %

Table 4.5: *Performance of SVM with and without the scaling step. Computational time for scaling each sample: 0,5 s aprox.*

Note that the program execution time can vary depending on which processor we use.

One against the Others (1vsAll)

One against the others (section 3.6) was the first strategy we used to perform the verification using SVM. So that, we construct 52 SVM each one using the pictures of one identity with label +1 and the rest with label -1 for the training. Once we had the models, the verification is done as it is shown in figure 4.6

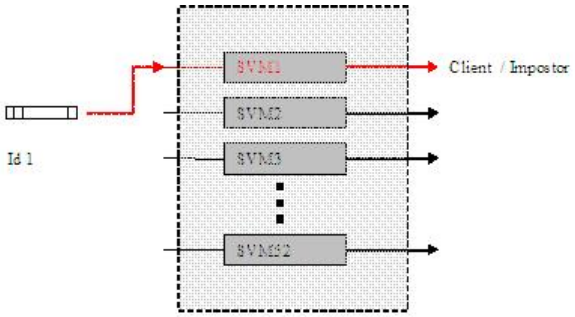


Figure 4.6: *Classification using 1vsAll strategy*

Using this strategy and the data of the MC scene described by BANCA, we took some measures of the performance using different kernels and different values of the rest of the parameters. In each case we have used the best C and γ obtained in the cross validation step.

Effect of the unbalanced data

Since for each SVM we have used 10 images of the client and 1560 of the impostor (MC scene), the training data is highly unbalanced. That

means that even having 90% of accuracy in the classification step our SVM can be completely useless (if we classify all the samples as impostors, the classification rate is great, but we do not reach the objective of verifying, that is the reason why we used the two measures FRR and FAR). In order to solve this problem we have used less impostor samples (100 chosen randomly) and we have weighted the importance of the client samples when constructing the models.

The next figure shows the problem we have described: with unbalanced data we score a 90% of classification accuracy, but almost half of the clients were rejected. Balancing the data we reduce almost 10% of the clients rejected.

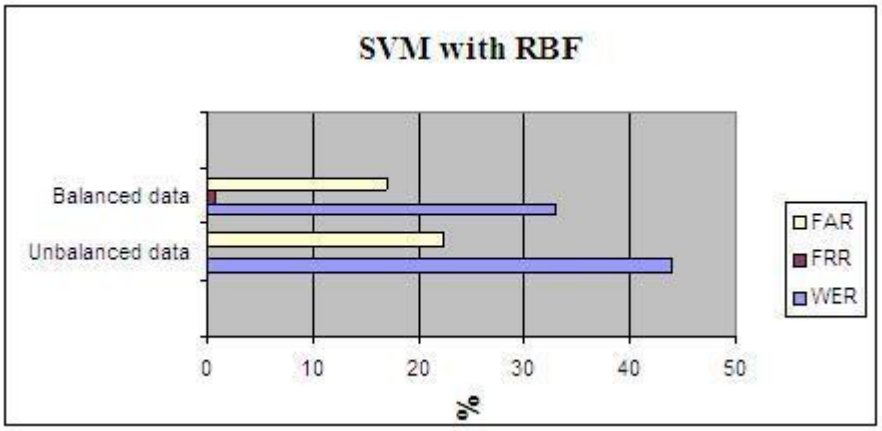


Figure 4.7: *Effect of the unbalanced data*

Using the models obtained with the balanced data, we have tested several kernels (also for MC scene) to verify which one performs the best. In the case of the polynomial kernel we have computed results for degrees 1 to 6. The results are shown in the graphics of figure 4.8 and figure 4.9.

This graphics show that the best performance is reached by Polynomial kernel (degree 1 and r=0) and sigmoidal kernel. However, if we take into account that in the case of sigmoidal kernel it is necessary a higher C to reach the same performance as polynomial kernel, then the best option is choosing the polynomial kernel (degree 1).

Moreover, it was longer to obtain the models in case of using Sigmoidal kernel than in the case of Polynomial kernel, because the cross validation step (for Sigmoidal kernel C is higher)

Note that for the two best kernels the results of FRR and FAR are very compensate, this property makes them to be suitable for a verification

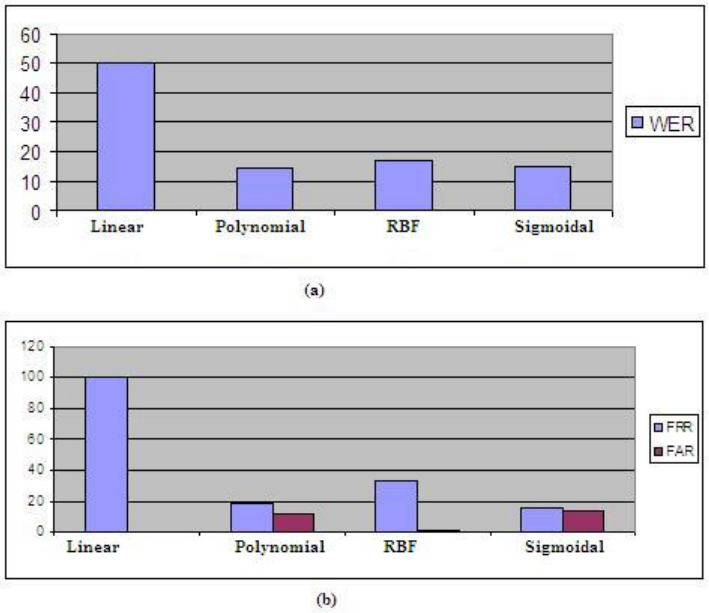


Figure 4.8: *In graphic (a) it is shown the WER obtained for each kernel. Graphic (b) shows FRR and FAR are specified.*

system.

The last experiment we did using *One against the Others* strategy was applying the complete BANCA protocol for the best case (Polynomial kernel degree 1 anr $r = 0$)in order to compare the results with the State of the Art ones [12] as we have done for the approach PCA+Euclidean distance-based classifier.

(%)	MC	UD	UA	P
FRR	17,69	49,42	30,70	2,90
FAR	11,3	8,11	29,92	13,32
WER(R=1)	14,5	28,76	30,31	8,11

Table 4.6: *Results obtained applying BANCA Protocol to the approach PCA + Euclidean distance-based Classifier*

One against One (1vs1)

In this section we will expose the results obtained while using the strategy One against One to perform the verification. The next diagram, figure 4.10,

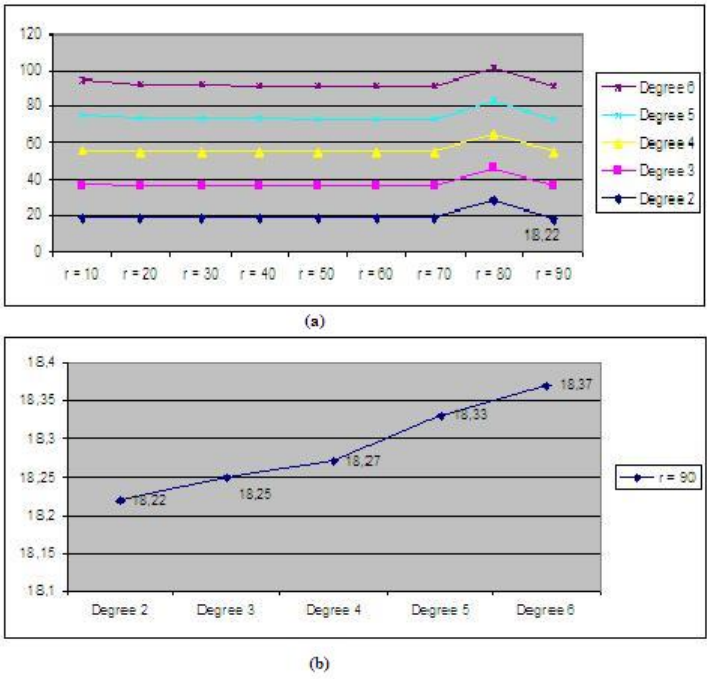


Figure 4.9: *In (a) it is represented the variation of WER while changing parameter r in the degree kernel. Finally, in (b) it is shown the performance of polynomial kernel varying the degree*

(*WER*%)	LDA+SVM	LDA+MLP	PCA+Euclidean	PCA+SVM(1vsAll)
MC	5,8	6,41	19,06	14,5
UD	34,55	15,62	37,88	28,76
UA	32,17	13,3	33,66	30,31
P	24,17	15,59	29,53	8,11

Table 4.7: *Comparative results between LDA+SVM, LDA+MLP, PCA+Euclidean and PCA + SVM(1vsAll) using polynomial kernel degree 1*

shows how is done the verification.

The a priori advantages that presented this strategy in comparison to the other were that it was not necessary to balance the data, because we used the same number of samples to train each SVM. Moreover, it creates a more accurate model of each identity in each of the SVM created. The disadvantages are clear: since we have to create one SVM for each pair

of identities the training time is longer. Moreover, if we want to include another identity in our system we have to retrain the whole system, while in the *1vsAll* approach it was not necessary.

The first we tried was computing WER, FRR and FAR for MC protocol and compare them to 1vsAll approach.

(%)	MC
FRR	15,28
FAR	35,11
WER(R=1)	25,2

Table 4.8: *Results obtained applying BANCA Protocol to the approach PCA + multiclass SVM (1vs1) with Polynomial kernel degree 1 and r=0*

Unfortunately, this results show this strategy is worst than 1vsAll in order to perform the classification, the reason should be that we are modelling each class very good for each SVM, but when we are verifying the identity of one sample the *impostor* model is not well-defined. Moreover, the com-

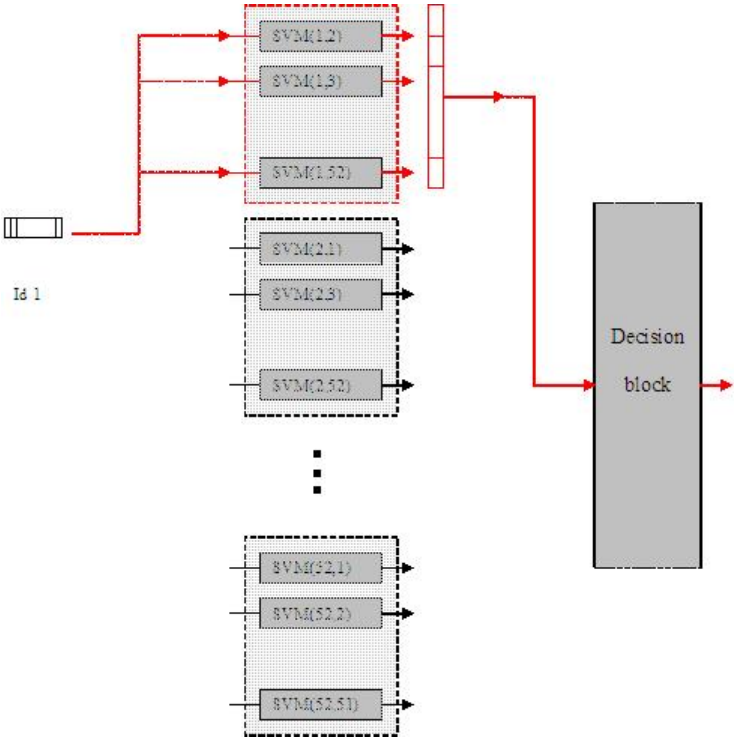


Figure 4.10: *Classification using 1vs1 strategy*

putational time used is almost double in comparison to the 1vsAll strategy. These are the reasons why we finally have concluded that 1vsAll is the best strategy to implement the classifier for a verification system. It is probable that for a recognition system 1vs1 is better than 1vsAll.

Results using the WM subset for training

In this part we have implement the classifier using 1vs All strategy, but using the identities of the WM dataset as impostors in the training set as it is shown in figure 4.11. To test the performance we have used the testing images of scene P.

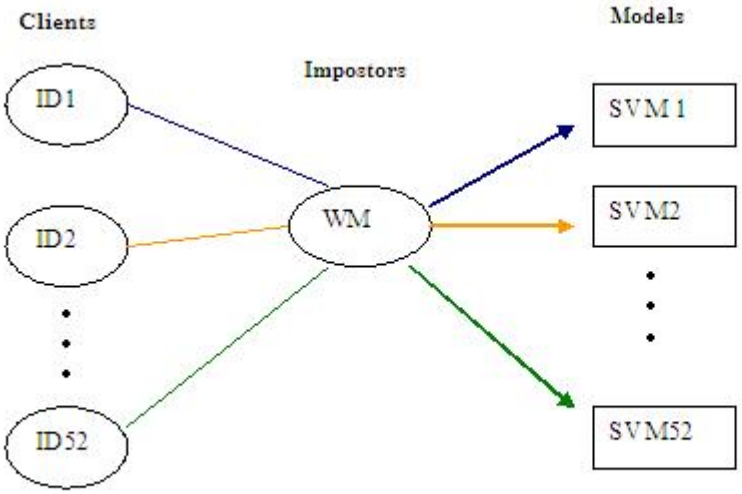


Figure 4.11: *Training of SVM using WM samples as impostors*

The following table shows the results.

(%)	WM trainingset and P testset
FRR	3,91
FAR	9,71
WER(R=1)	6,81

Table 4.9: *Results obtained applying BANCA Protocol to the approach PCA + multiclass SVM (1vs1) with Polynomial kernel degree 1 and r=0*

As we can see in table 4.9 the results we get using WM dataset in training are better, but the best is that when using this method it is possible to add other identity to the system in an easy way: we only have to train other SVM, otherwise we should have had to re-train the last 52 SVM.

Chapter 5

Conclusion and Future Work

In our experiments all the pictures we have used to test all the algorithms of verification and/or recognition we have built are from BANCA database, so that we have compared the results obtained with the ones from the State of the Art which have used BANCA as well.

PCA is a good technique to reduce dimensionality, and if we use it with a distance-based classifier we get acceptable results (using Euclidean distance in the classifier, since Mahalanobis distance does not perform well). But this technique of classification is the most basic we can use, in order to improve the results we have tested several configurations of multiclass-SVM.

The first configuration we tested was was the so called 1vsAll. The results obtained with this configuration of SVM, and using PCA to reduce dimensionality, are similar to the ones obtained using LDA and SVM that we found in the literature, but we clearly improve them, and the computational cost is lower using PCA than the one using LDA. This results mean that the previous supposition of that the maximization of the discriminatory information of LDA was not going to improve the final results was correct, SVM works really well finding the optimal separating hyperplane and it compensates the fact that PCA does not consider the class separability. However, while comparing the techniques we also can see that the results obtained using MLP and SVM are the best.

Using the second configuration, the so called 1vs1, we obtained worse results than the ones obtained using 1vsAll. The reason should be that when using this technique we are creating very good models of each client but we do not have a correct model of an impostor while if we use 1vsAll we create for each SVM a model of a client and a model of an impostor.

The conclusion, then, is that PCA is a good technique to perform the

reduction of dimensionality, because it decreases the classification complexity when using SVM in the classification task. For it, the configuration of 1vs1 should work very well for recognition, when having a good model of each identity is desirable, but for verification, 1vsAll performs better and it is necessary less computationally cost to train it than for the other configuration.

SVM is a powerful technique that has allowed us obtaining very good results for face verification, but the performance they offer can be better. There are several configurations of them that we have not tested in the present work that can imply an improvement respect the results we obtained. One example of a new configuration of the SVM can be MSVM (Mixture of SVM) which consist of training N SVM (chosen randomly) in a first layer and after train a second layer of SVM on the margins. Moreover, applying this technique it is possible to decrease the computational complexity of the classification problem, which presented a problem during the training. But the line of research about SVM is not limited on this, nowadays there is an opened research about new kernels to use with SVM.

Appendix A

SVM library

SVM (Suport Vector Machine) is a new technique for data classification. Although being considered easier to use than Neural Networks, users who are not familiarized with this technique can obtain unsatisfactory results. So that, this library intends to solve the difficulties that can be found at the beginning and give some recipes to the user to obtain acceptable results fast and easily.

A.1 Introduction

Although the general theory of SVM has been explained in detail in **??**, we briefly recover some SVM basics which are necessary for explaining the whole procedure of the library.

A classification task usually involves with training and testing data which consist of some data instances. Each instance in the training set contains one "target value" (class labels)and several "attributes" (features). The goal of SVM is to produce a model which predicts target value of data instances in the testing set which are given only the attributes.

Given a training set, the Support Vector Machines require the solution of the following optimization problem:

$$\min_{\omega,\mathrm{b},\xi} \tfrac{1}{2}\omega^T\omega + C\sum_{n=1}^{l}\xi_i$$

$$\text{subject to } y_i(\omega^T\phi(x_i) + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0.$$

Training vectors x_i are mapped into a higher dimensional space by function ϕ. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term.

Besides, $K(x_i, y_i) \equiv \phi(x_i)^T \phi(x_j)$ is called the kernel function. Although new kernels are being proposed by researchers, the four basic kernels are:

$$\text{-Linear: } K(x_i, y_i) = x_i^T x_j.$$

$$\text{-Polynomial: } K(x_i, y_i) = (\gamma x_i^T x_j + \tau)^d, \gamma > 0.$$

$$\text{-Radial Basis Function(RBF): } K(x_i, y_i) = exp(-\gamma \|x_i^T x_j\|^2), \gamma > 0.$$

$$\text{-Sigmoid: } K(x_i, y_i) = tanh(\gamma x_i^T x_j + \tau), \gamma > 0.$$

Where,γ, τ and d are kernel parameters.

A.1.1 Procedure

The proposed procedure by the authors is schematized as follows:

- Transform data to the format of an SVM software.
- Conduct a simple scaling on the data.
- Consider one of the kernels.
- Use cross-validation to find the best parameter C and gamma.
- Use the best parameter C and γ to train the whole training set.
- Test.

A.2 Data Preprocessing

SVM requires that each data instance is represented as a vector of real numbers. Thus, if there are categorical attributes, it is necessary to convert the into numeric data. For example, in the case of a three-category attribute, such as red, green,blue can be represented as (0 0 1),(0 1 0) and (1 0 0).

Scaling the data before applying SVM is also very important. The main advantadge of doing it, is to avoid attributes in greater numeric ranges dominate those in smaller ranges. Another advantage is to avoid numerical difficulties during the calculation: large attribute values can cause numerical problems, because kernel values usually depend on the inner products of feature vectors, e.g. the linear kernel and the polynomial kernel. So that, it is recommended to scale linearly each attribute to the range [-1,+1] or [0,1].

A.3 Training and prediction

The first step to use the SVMlib is deciding which kernel do we use as well as the penalty and kernel parameters.

Let's suppose we choose the RBF Kernel. The reason for this choice is that this kernel nonlinearly maps samples into a higher dimensional space, so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear. Furthermore, the linear kernel is a special case of RBF (the linear kernel with a penalty paramenter C has the same performance as the RBF kernel with come parameters (C,γ). In addition,the sigmoid kernel behaves like RBF for certain parameters.

The next step is choosing the correct parameters of the kernel to perform well. For this it is recommended using cross-validation and other tools included in lib-SVM that are further explained in [14].

Once the model is correctly created, it is possible to apply the function svm-predictor to classify the samples of the chosen testing set.

Appendix B

Kuhn-Tucker Theorem

In this Annex, we will expose the Kuhn-Tucker Theorem, which is used in the development of the Support Vector Machine Theory.

Given an optimization problem with the convex domain $\Omega \subseteq R^N$

minimize $f(z)$ $z \in \Omega$ subject to

- $g_i(z) \leqslant 0$ $i = 1...k$
- and $h_i(z) = 0$ $i = 1...m$

With $f \in C^1$ convex and g_i,h_i affine, necessary and sufficient conditions for a normal point z^* to be an optimum are the existence of α^*, β^* such that

1.
$$\frac{\partial L(z^*, \alpha^*, \beta^*)}{\partial z} = 0 \tag{B.1}$$

2.
$$\frac{\partial L(z^*, \alpha^*, \beta^*)}{\partial \beta} = 0 \tag{B.2}$$

where $L(z, \alpha, \beta) = f(z) + \sum_{i=1}^{k} \alpha_i g_i(z) + \sum_{i=1}^{m} \beta_i h_i(z)$

3.
$$\alpha_i^* g_i(z^*) = 0 \tag{B.3}$$

4.
$$g_i(z^*) \leqslant 0 \tag{B.4}$$

5.

$$\alpha_i^* \geqslant 0 \tag{B.5}$$

The third condition is known as the Karush-Kuhn-Tucker complementary condition. It implies that for active constraints $\alpha_i \geqslant 0$ and for inactive constraints $\alpha_i = 0$. That condition is the one which allow us to identify the training samples that define the largest margin hyperplane (Support Vectors)

Bibliography

[1] S.Bengio, F.Bimbot, J.Mariéthoz, V.Popovici, F. Porée, E.Bailly-Baillière, G.Matas, B.Ruiz, and J.P.Thiran, "The banca database and evaluation protocol," *4th International Conference on Audio- and Video-Based Biometric Person Authentication, Surrey, UK.*, 2003.

[2] He, Yan, Hu, Niyogi, and Zhang, "Face recognition using laplacianfaces," *IEEE transactios on Pattern analysis and machine intelligence*, March 2005.

[3] L.Wiskott, J. andN, Krüger, and C. D. Malsburg, "Face recognition by elastic bunch graph matching," *IEEE Transactions on Pattern analysis and machine intelligence*, July 1997.

[4] S. Ripley, "Pattern recognition and neural networks," 1996.

[5] R. Duda, P. Hart, and D. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[6] R. Chellappa, K. Fukushima, A. Katsaggelos, S.-Y. Kung, Y. LeCun, N. M. Nasrabadi, and T. A. Poggio, "Applications of artificial neural networks to image processing (guest editorial)," *IEEE Transactions on Image Processing*, vol. 7, pp. 1093–1097, August 1998. (original is digital pdf).

[7] B. Ripley, *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press, 1996.

[8] P. Belhumeur, J. P. Hespanha, and DJ.Kriegman, "Eigenfaces vs fisherfaces: Recognition using class specific linear projection," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, 1997.

[9] Liu and Wechler, "Evolutionary pursuit and its application to face recognition," *IEEE transactios on Pattern analysis and machine intelligence vol.22*.

[10] www.itl.nist.gov/iad/humanid/feret.

[11] www.tele.ucl.ac.be/PROJECT/M2VTS.

[12] Marcel, "Face verification using lda and mlp on the banca database," *Technical Report IDIAP-RR 03-66*, 2003.

[13] Chih-Wei, H. Chih-Chung, Chang, and C.-J. Lin, "A practical guide to support vector classification," *4th International Conference on Audio- and Video-Based Biometric Person Authentication, Surrey, UK.*

[14] www.csie.ntu.edu.tw/ cjlin/libsvm.